

Technological Feasibility Analysis

1 November 2021



Team Truthseeker

*Garry Ancheta
Georgia Buchanan
Jaime Garcia Gomez
Kyler Carling*

Project Sponsor

NOBL Media – Jacob Bailly

Team Faculty Mentor

Felicity H. Escarzaga

Table of Contents

1- INTRODUCTION	1
2 - TECHNOLOGICAL CHALLENGES	3
3 - TECHNOLOGICAL ANALYSIS	4
USER INTERFACE STACK	5
DATA VISUALIZATION	12
USER AUTHENTICATION	22
API DESIGN	30
4 - TECHNOLOGICAL INTEGRATION	37
5 - CONCLUSION	40
6 - REFERENCES	41



Chapter 1 - Introduction

Misinformation is profitable. There is always a business entity using misinformation for profit. In the 50s, tobacco companies purported that cigarettes were medicinal; they created advertisements proclaiming that physicians believed in cigarettes' "medicinal" benefits [1]. The tobacco industry hid the truth from consumers for as long as it could. Still, the United States Surgeon General's report in 1964, showing that smoking cigarettes were harmful, became the pivotal point of the shifting public opinion on smoking cigarettes [2]. With an all-time high in taxes against tobacco companies [2], cigarette smoking is at its all-time lowest profitability [2].

Today, misinformation is widespread on the internet. Different platforms intentionally spread misinformation to harm individuals and groups. The internet, however, is just another representation of many businesses through websites. For most businesses, websites are another source of income through the showing of advertisements. Demand Site Platforms (DSPs) are what allow the placement of ads on websites. Currently, DSPs deal with misinformation by blacklisting or demonetizing websites. However, DSPs only do so when they are actively told to by the advertiser. Thus, by being associated with misinformation, an advertiser's reputation is damaged. A damaged reputation is a waste of money since it might deter customers from supporting the advertiser when the advertiser is supporting misinformation, intentional or not. This is where the Misinformation and Credible News Analysis Tool comes in. The tool allows advertisers not to support businesses that spread misinformation by rating how credible a page is on a website. Using this credibility score, the tool can decide whether an advertisement should be placed on that page based on its credibility. Unlike the current process where businesses must tell DSPs which websites the advertisers' ads should not be placed on, NOBL Media has the capability to automate the process by allowing advertisers to simply tell NOBL Media what credibility rating a website should have for their advertisements to be placed.

Problem

NOBL Media collects advertisement information for advertisers who choose to use NOBL Media's services. Once that information is collected, advertisers can see how their advertisement is performing in terms of supporting misinformation and



how much money the advertiser is losing due to misinformation. Currently, NOBL Media's business flow is interrupted by the following problems:

1. NOBL Media customers do not have a way to visualize the data NOBL Media collects about the customers' advertisements.
2. NOBL Media cannot abstract the data so that it is easily understood by customers.
3. NOBL Media does not have a way to directly give the customers' advertisement data.

These problems remove NOBL Media's ability to show proof that their service works for the customer. As a result, NOBL Media would be unable to retain current customers and attract new customers.

Solution

To solve the project's problems, NOBL Media requires two components: the web application and an Application Programming Interface (API). These two components allow NOBL Media to solve the problem with their business flow through the implementation of the following features:

1. The web application must be able to abstract data coming from the NOBL Media database and represent these to customers through graphs and charts.
2. The web application must be able to create and authenticate customer accounts to allow secure access to the customer's data.
3. The web application must allow customers to download customer ad data in a formatted file such as in a CSV or Excel file.
4. The API must be able to retrieve data from the NOBL Media database and send this data to the web application.
5. The API must be able to handle authentication requests from the web application.



Chapter 2 - Technological Challenges

Within the solution's components are challenges that need to be addressed to allow a cohesive structure to make the user experience to be as smooth as possible and ultimately achieve the goal of allowing users to see the proof of NOBL Media's service working as intended. The web application component has challenges relating to the user interface and data visualization. The API component has a challenge relating to its design and capabilities. In detail, the challenges are as follows:

a) Challenge #1 - User Interface Stack

Ensures that the project is responsive and interactive to use by NOBL Media's users.

b) Challenge #2 - Data Visualization

Ensures that the project can show charts containing statistics on each NOBL Media user's advertisement campaigns.

c) Challenge #3 - User Authentication

Ensures that user information is protected and prevents unauthorized access to advertisement campaign data.

d) Challenge #4 - API Design

Ensures that the project can rapidly retrieve data from NOBL Media's database and be able to send this data to the web application.



Chapter 3 - Technological Analysis

This section goes into the details surrounding each technological challenge identified for the project. As more information is presented for each technological challenge, the solution for each will be uncovered. This section will discuss why this challenge is a challenge, what ideal characteristics the solution needs for the challenge, each candidate that is a possible solution, how each candidate holds up in each characteristic, and finally, explain the chosen candidate.



SECTION 3.1 - User Interface Stack

Introduction

A user interface stack is a combination of three different technologies: UI library language, the UI library, and the rendering framework. The Misinformation and Credible News Analysis Tool's web application will implement an interface stack to serve present campaign data.

The Misinformation and Credible News Analysis Tool's UI stack language is TypeScript. TypeScript allows developers to define data types, such as integers or characters, to catch errors at compile time [4]. This is incredibly important since it can reduce the chances of bugs (unintended functions and errors) that occur when the project is fully built and ready for user interaction. TypeScript is the main dependency of the other two technologies in this stack.

Website UI libraries allow developers to focus on the high-level functionality of the website rather than the low-level functionality. In other words, website UI libraries provide reusable tools for developers to use, so they do not have to constantly create their own tools. This allows more time for design and implementation of the web application's features instead of building the interface of the web application itself.

Ideal Characteristics

Complexity

Complexity refers to the learning curve of the UI library based on features. The UI Library candidate must be modular and minimalist. This means that the UI Library must allow for components of the website to be built and reusable in different pages. For example, in an ordinary website built with HTML and JavaScript, a page within the website might have a complex gallery of pictures; to reuse this gallery in another page, all the code must be copied and put into that page



once more. This makes the code for pages larger and ultimately unreadable as more functionality is added within a page. Modularly and minimalist UI Libraries allow for the creation of this gallery page and package it as a component, which means that any time that it is needed to put in different pages, all it takes is to call the component instead of copying all the code.

Easy startup refers to how easy it is for developers to spend time learning the UI library proprietary functions rather than relying on programming fundamentals. Easy startup is preferred since it saves development time for the Misinformation and Credible News Analysis Tool's other features. This characteristic is rated between 1 to 3 points total with each point signifying the following:

1 point - complex programming, mastery of requirements is required to create simple parts of the stack.

2 points - Allows for the creation of components, easy to understand documentation in one of the two components of the stack.

3 points - Allows for the creation of components, easy to understand documentation, and provides direct understandable implementation of the library's features in both components of the stack.

Client Preference

The client's preference for which website UI library to use. NOBL Media's preference is considered because NOBL Media will be maintaining the product after the project is finished. This characteristic is rated between 1 to 2 points total with each point signifying the following:

1 point - Client does not prefer the stack

2 points - Client prefers the stack

Rendering Methodology

This characteristic refers to the stack's method of rendering websites. Currently, websites are rendered in two different ways: client-side and server-side rendering. Client-side rendering allows a user to load all parts of a website in the initial load



without having to load more whenever the user switches to different pages of the website. Server-side rendering allows a user to load parts of a website only when the user goes to that specific website.

The project, in the opinion of the client, would be more suited with client-side rendering. Client-side rendering allows the users of the web application to easily go to different pages without having to worry about load times. This is especially useful when a user logs in and goes to their ad campaign's page because it allows only the data visualization to be the only thing that needs to be loaded when the user visits the page. In comparison, server-side rendering would make the user load the entirety of the ad campaign's page and not just the data visualization which could negatively affect the user's experience especially when the loading time is long. The rating for this characteristic is as follows:

1 point - uses server-side rendering

2 points - uses client-side rendering

Candidates

The candidates consist of two components: the UI library and the rendering framework. There are two candidates in total separated between two rendering framework libraries. Within these UI libraries are the rendering frameworks that are made specifically for either client-side or server-side rendering. The two candidates are React & Gatsby, React & NextJS.

React & Gatsby

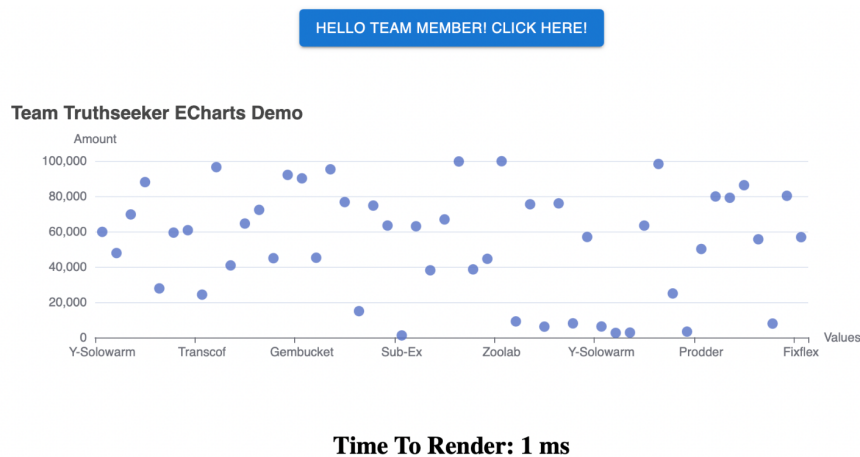
React is by far the most popular UI library today. React is an open-source front end UI library maintained by Facebook and other community developers and companies and was initially released in May 2013 [5]. It has a minimalist design philosophy and is also community driven.

Gatsby is the leading framework for client-side rendering for React made by a company of the same name [6]. While a company technically owns Gatsby, it is also open-source, actively built by a community [6]. Gatsby is used by different well-known companies such as Snapchat, Tinder, Giphy, and many more [6].

Complexity

React is considered a minimalistic UI library. Whenever a developer needs additional features there are a lot of third-party apps to include from the React community. This also makes react modular. Another distinctive feature of React is that it uses JSX which is a syntax extension of JavaScript. JSX allows JavaScript/TypeScript and HTML to be written together in the same file. Documentation for React is understandable and since React is community-driven, it means that React has a lot of tutorials on the internet to choose from for learning purposes.

Gatsby's focus is to allow the creation of websites to be direct and manageable [7]. This makes integrating Gatsby into the project much easier as setting up Gatsby is straightforward. Referring to Figure 3.1.1, this is a simple web page created with React and Gatsby. Its creation only took an hour due to React and Gatsby's low complexity. Due to React and Gatsby's shallow learning curve, detailed documentation, and strong community support, React and Gatsby scores 3 out of 3 points in this characteristic.



Time To Render: 1 ms

Figure 3.1.1 – Simple React & Gatsby Webpage



Client Preference

NOBL Media's Chief Technology Officer, Jacob Bailly, has stated his preference for React and client-side rendering. Jacob Bailly's preference is important because NOBL Media will be maintaining the application in the future. Due to the client's preference for this stack, this stack is rated 2 out of 2 points in this characteristic.

Rendering Methodology

Gatsby is a client-side rendering framework. Additionally, Gatsby also has different features which allow it to load pages faster such as deferred static generation, which allows a page to load only the critical parts of the page, and selective content preloading, which allows developers to tell a page which order it should load the page's contents. [6]. Since this is NOBL Media's preferred rendering methodology and Gatsby's added benefits of selective content preloading and deferred static generation, the stack rates 2 out of 2 points in this characteristic.

React & NextJS

NextJS is the leading framework for server-side rendering for React made by Vercel Inc [8]. NextJS is used by different well-known companies such as Twitch, Starbucks, Netflix, GitHub, and many more [8].

Complexity

While React has been established to have low complexity, NextJS has vivid documentation and can be initialized without any complicated steps. However, the biggest problem with this stack is not the framework itself but due to its rendering methodology. Server-side rendering can become complex as the website gets built with more and more functionality [9]. As a result, this stack is rated 2 out of 3 points in this characteristic.



Client Preference

The client does not prefer this stack since NextJS is a server-side rendering framework. Therefore, this stack rates 1 out of 2 points in this characteristic.

Rendering Methodology

NextJS is a server-side rendering framework which means that all pages are loaded only when the user intends to go to a specific page. This is not ideal for the web application due to its need to cater to the user and one of the ways the web application intends to cater to the user is to make loading the website as fast as possible. With all the data that the web application will need to handle such as user account data and more importantly, the user's ad campaigns' data, server-side rendering would be inconsistent with loading times. In comparison, client-side rendering loads every page when the user arrives at the website which makes loading pages consistent. Due to server-side rendering's inconsistent loading times and increasing complexity as the web application's complexity increases, the stack rates 1 out of 2 points for this characteristic.

Chosen Candidate

Referring to Table 3.1.2, React & Gatsby is the chosen candidate since the stack is preferred by the client and the rendering methodology is what the project needs. In both cases, React & Gatsby scores 1/1 in the rating. React & NextJS is hindered by the client preferring client-side rendering rather than server-side rendering. Additionally, server-side rendering's increase in complexity over time makes it harder to maintain as the web application obtains more features.

	Complexity	Client Preference	Rendering Methodology	Total
React & Gatsby	3/3	2/2	2/2	7/7
React & NextJS	2/3	1/2	1/2	4/7

Table 3.1.2 – Candidate Ratings



Proving Feasibility

Proving feasibility for this solution requires a prototype of a website while using the language TypeScript. Additionally, the prototype would also show different functions that Gatsby provides such as deferred static generation and selective content preloading.



Section 3.2 - Data Visualization

Introduction

One of the most significant aspects of the Misinformation and Credible Analysis Tool is visualizing the data from the backend. The API component should handle many identifiers for this data, such as grabbing the data specifically for one user. However, there are different aspects of how the web application will visualize the data for the user. The most important part of data visualization is that it is understandable. Furthermore, depending on the user's needs, the product's data visualization needs to have different visualization methods. Thus, whichever data visualization library that the project uses must be extensive enough to accommodate the current and future data visualization requirements. The current data visualization requirements range from bar charts to pie charts to scatterplots.

Ideal Characteristics

The ideal solution for data visualization would allow large quantities of data points in a single graph/chart and render charts rapidly. For a more comprehensive solution for this challenge, the characteristics that the project requires is as follows:

Data Handling

Data handling should be fast so that the website can build and create the data visualization at will. Additionally, the data visualization library should also handle large amounts of data points since a user might have a substantial amount of data for different ad campaigns. This is important to the project due to the possibility that NOBL Media customers may want to look at an extreme amount of data points in addition to not needing to wait for charts to take a long time to create the data points. The scoring of this characteristic is rated out of 3 points with each score being described as follows:



1 point – Unable to handle more than 5000 data points and/or cannot process and render data points under 3 seconds.

2 points – Unable to handle more than 25,000 data points and/or cannot process and render data points under 2 seconds.

3 points – Unable to handle more than 50,000 data points and/or cannot process and render data points under 500 milliseconds.

4 points – Unable to handle more than 75,000 data points and/or cannot process and render data points under 250 milliseconds.

Interactivity

Interactivity refers to the data visualization library's ability to create a visualization that the user can interact with. This is important due to the project's need for the user to see their ad campaign's data in as many ways as possible; so, the user having the ability to change what the chart is showing or seeing more information about the chart helps NOBL Media service the users' needs. An example of this would be a user hovering over each bar within a bar chart and learning more specific information on that data point. The scoring of this characteristic is rated out of 2 points with each score being described as follows:

1 point – Unable to have any interactivity with the chart at all after rendering.

2 points – Interactivity is limited to zooming into the chart and changing the amount of data points being shown.

3 points – Hovering and clicking on data points on the chart has functions such as a tooltip popping up. Additionally, the chart can contain dynamic data, meaning that the user is able to change the amount of data points being shown on the chart without zooming in.

Candidates

There are four candidates for the project's data visualization. All candidates are component-based, meaning that the candidates produce graphs so that each graph can be placed on a website as an HTML tag; this makes the structure of



the web application easier to understand which helps with the longevity of the web application component of the project. The candidates are ChartJS, Ant Design Charts, ECharts, and Recharts.

ChartJS

ChartJS is one of the most popular libraries for data visualization boasting a beginner-friendly usage [11]. ChartJS is unique in comparison to most data visualization libraries in that it is crowdsourced, meaning that ChartJS is not owned by one company, but is built by different individuals willing to contribute towards the library [11]. ChartJS was found through a search in npmjs for the most popular data visualization libraries [11].

Data Handling

ChartJS uses Canvas API to be fast in rendering charts in comparison to Flash and SVG-based data visualization libraries. Since ChartJS is Canvas-based, it can also handle a lot of data, making it ideal for the project. ChartJS has 9 basic types of charts but can be highly customized so that many chart subtypes can be created [11].

Referring to Figure 3.2.1, the team was able to produce a mock chart with 100,000 data points. ChartJS was able to render the entire graph in 2489 milliseconds as seen on the top left portion of the chart; in seconds, this would be about 2.5 seconds. While this might be slow, it does show that ChartJS is able to handle an extreme amount of data points.

Team Truthseeker ChartJS Demo

Time To Render: 2430 ms

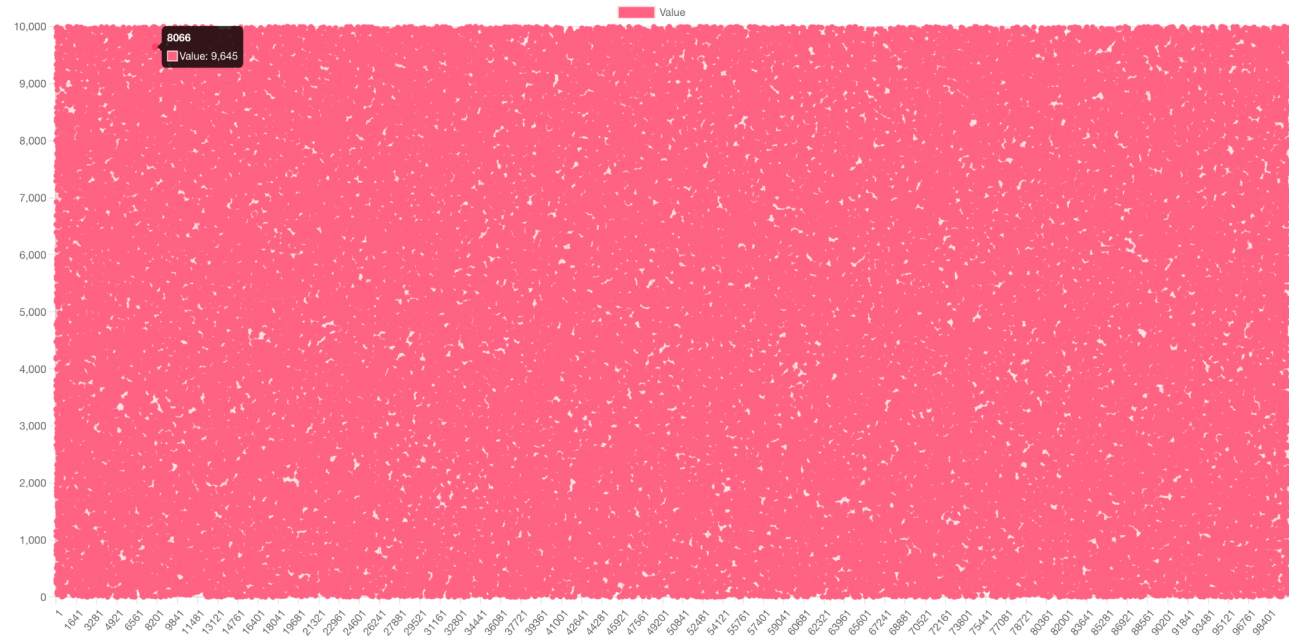


Figure 3.2.1 – ChartJS Demo with 100,000 data points

Referring to Figure 3.2.2, ChartJS could rapidly render the chart with 5,000 data points since looking at the top left of the figure, the chart was rendered in 315 milliseconds. This shows ChartJS' ability to render quickly when it comes to lower data points. While ChartJS could render 5,000 data points rapidly, ChartJS has trouble rendering the 100,000 data points rapidly and as a result, ChartJS scores a 3 out of 4 points in this characteristic.

Team Truthseeker ChartJS Demo

Time To Render: 315 ms

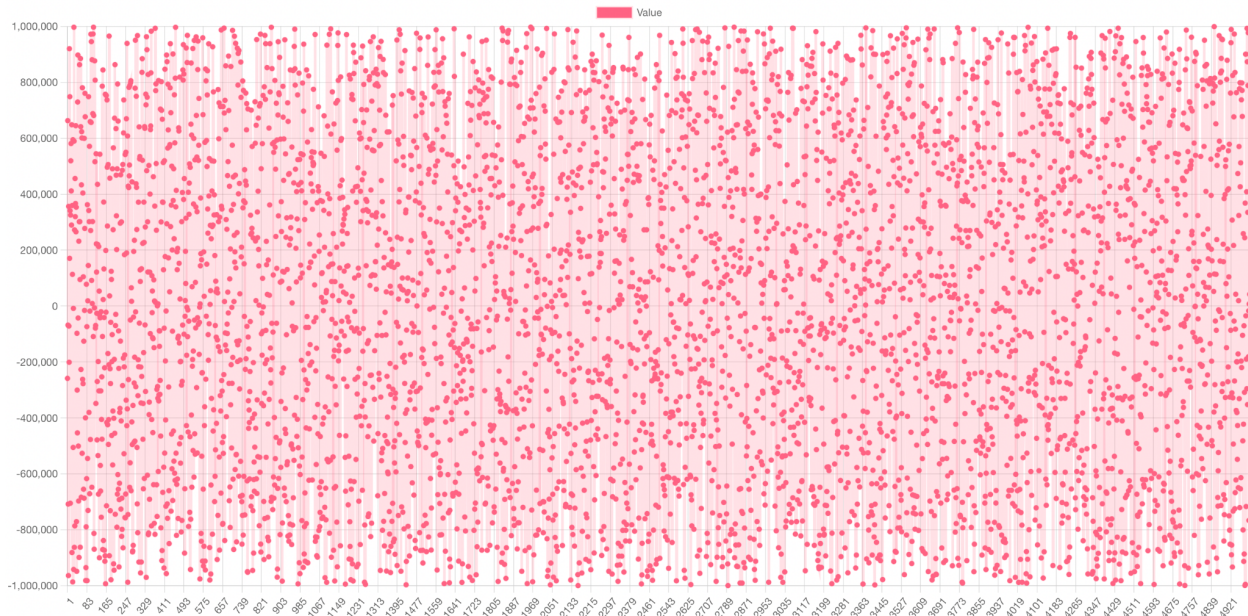


Figure 3.2.2 – ChartJS Demo with 5,000 data points

Interactivity

ChartJS has interactivity implemented within all chart types. In addition, ChartJS also has responsive capabilities for each chart depending on the user's window size and even when the user changes the window size. ChartJS also incorporates mobile usage and touchscreen functionality with their graphs. Referring to Figure 3.2.1, a tooltip can be seen showing the value of a data point. The functionality to hover over a data point and having a tooltip be revealed creates a level of interactivity that allows users to have the best experience when viewing data on the web application component of the project. Overall, ChartJS encompasses everything that the project would need in terms of interactivity. Due to ChartJS' high level of interactivity, ChartJS scores a 3 out of 3 points in this characteristic.

Ant Design Charts

Created about 2 years ago by the Chinese company Ant Design, Ant Design Charts is a semi-popular data visualization library. One of the biggest advantages of using Ant Design Charts is that it is component-based, meaning that the

different types of charts offered by Any Design Charts can be used as HTML tags. For example, a Line chart can be created by using the <Line> tag.

Data Handling

Ant Design Charts is SVG-based which is not as fast in terms of rendering as Canvas-based libraries. This can be shown in the following figure.

An attempt was made to render 100,000 data points onto one chart and the rendering took way too long that it was not completed at all. Referring to Figure 3.2.3, the data visualization library was able to handle 5,000 data points. However, in comparison to ChartJS, the first candidate analyzed, the time it took to render is significantly slower. At the top left of the figure, the rendering took 2113 milliseconds. Since Ant Design Charts could not render 100,000 data points and took about 2 seconds to render 5,000 data points, Ant Design Charts scores a 1 out of 4 points in this characteristic.

Time To Render: 2113 ms

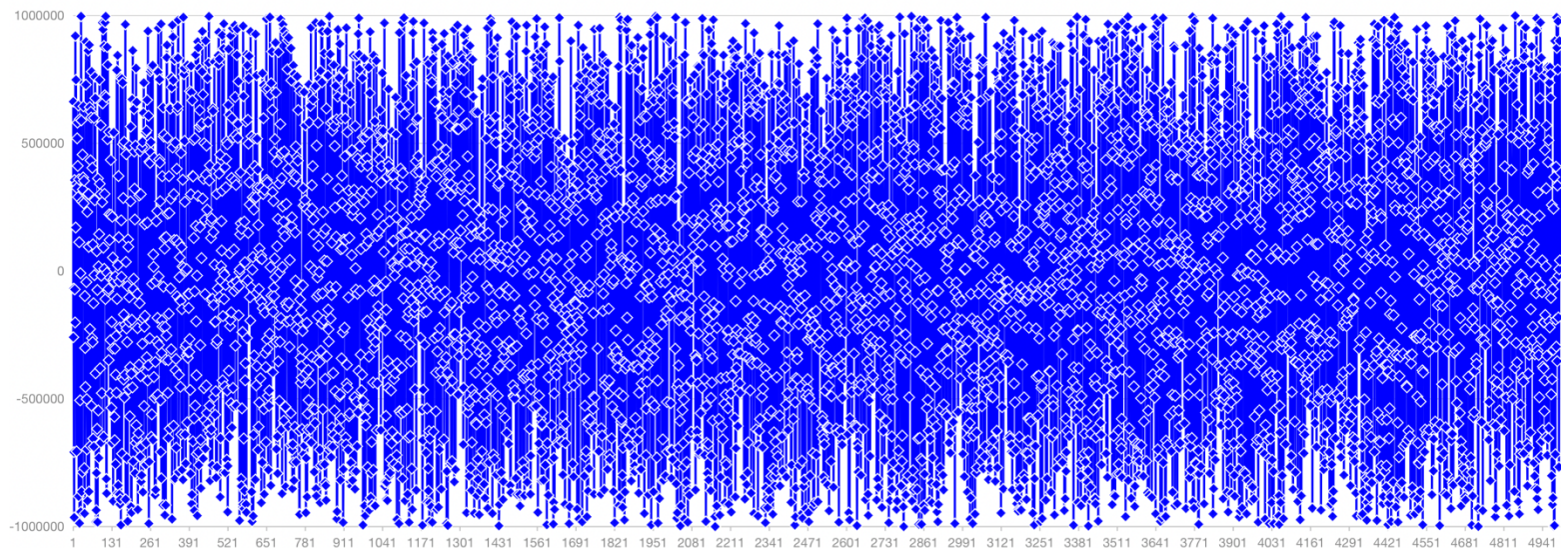


Figure 3.2.3 – Ant Design Charts with 5,000 data points

Interactivity

Ant Design Charts has the capabilities of interactivity ranging from zooming in to being able to bring a tooltip, showing more information about a data point. Additionally, Ant Design Charts makes customizing different parts of interactivity



easy to implement. Since Ant Design Charts has a lot of ways to implement interactivity, Ant Design Charts scores a 3 out of 3 points in this characteristic.

ECharts

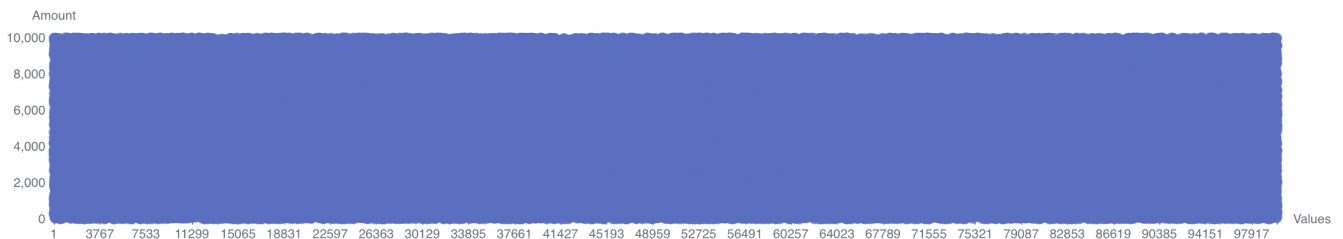
Fully known as Enterprise Charts, ECharts was created by the Chinese company Baidu. Considered to be one of the most powerful data visualization libraries out there, ECharts boasts a weekly download amount of about 71,000 [10]. ECharts is also used by many well-known companies such as Amazon, Gitlab, and Tencent.

Data Handling

ECharts is the only candidate that allows for both SVG-based and Canvas-based rendering. Furthermore, ECharts is one of the most powerful data visualization libraries released to the public which is proven by the following figures.

Referring to Figure 3.2.4, ECharts was able to render 100,000 data points in 195 milliseconds. In comparison to ChartJS, ECharts was able to render 100,000 data points **12.75x** faster. Referring to Figure 3.2.5, ECharts was able to render 5,000 data points in 100 milliseconds. This is **3x** faster than ChartJS and **21x** faster than Ant Design Charts, fortifying ECharts ability to be faster than the previous candidates. Due to ECharts' ability to render 100,000 and 5,000 data points in less than 200 milliseconds, ECharts scores 4 out of 4 points in this characteristic.

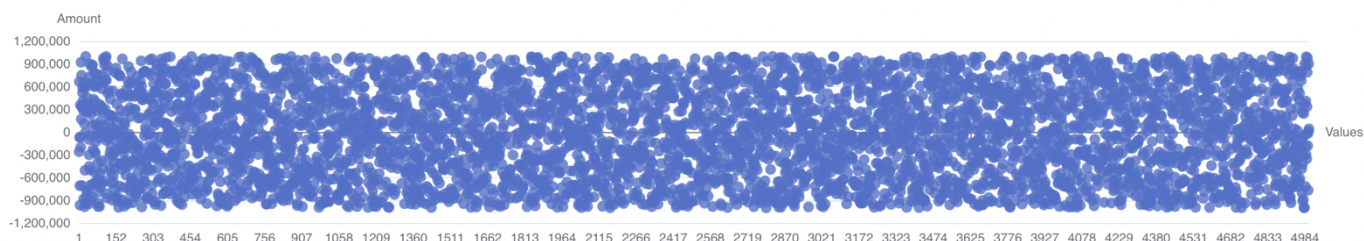
Team Truthseeker ECharts Demo



Time To Render: 195 ms

Figure 3.2.4 – ECharts Demo with 100,000 data points

Team Truthseeker ECharts Demo



Time To Render: 100 ms

Figure 3.2.5 – ECharts Demo with 5,000 data points

Interactivity

ECharts' interactivity is an important part of the library since it has different interactions that are not implemented in other data visualization libraries. For example, ECharts can create a graph where data points can be dragged. Additionally, it is easy to implement these interactions since the library abstracts them so that it can be added just by setting a certain variable true. Due to ECharts' edge in more interactions and easy implementations for these interactions, ECharts' scores 3 out of 3 points in this characteristic.

Recharts

Originally created in 2016, Recharts is a React-based data visualization library created by the Recharts Group [12]. Recharts is based on D3, another data visualization library [12]. This also means that Recharts is SVG-based meaning that its rendering capabilities are not as fast as Canvas-based data visualization libraries [12].

Data Handling

Recharts, being SVG-based, was not able to render 100,000 data points much like Ant Design Charts. However, Recharts was able to render 5,000 data points much faster than Ant Design Charts. Referring to Figure 3.2.6, Recharts was able to render 5,000 data points in 1711 milliseconds. In comparison, Ant Design Charts was able to render 5,000 data points in 2113 milliseconds. This solidifies Recharts to have much better data handling capabilities in comparison to Ant Design

Charts. Due to Recharts' inability to render 100,000 data points and its , Recharts scores a 2 out of 4 points in this characteristic.

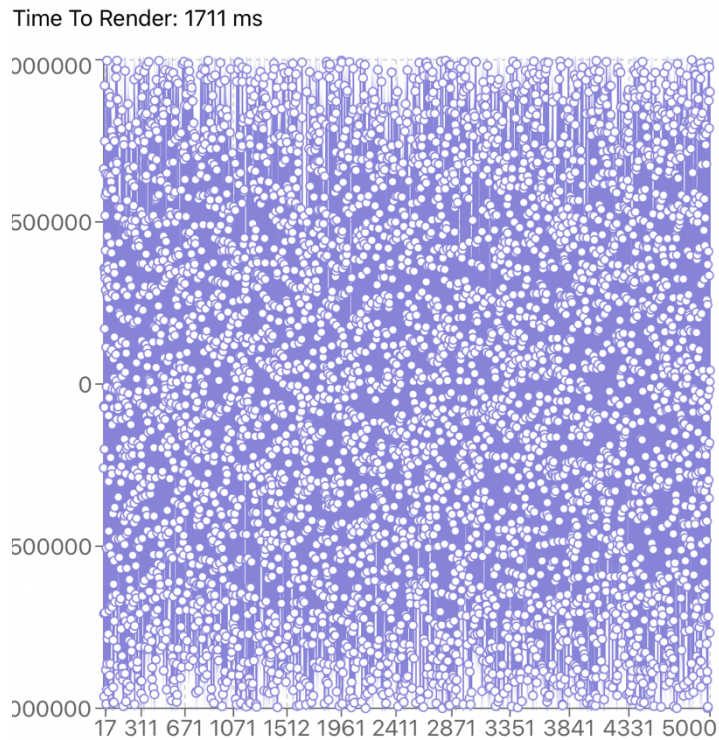


Figure 3.2.6 – Recharts Demo with 5,000 data points

Interactivity

Recharts has great capabilities for interactivity. What makes Recharts shine over the other candidates in its interactivity is that the tooltips for Recharts can be customizable beyond the extent of other candidates. For example, Recharts allows tooltips to have additional comments within it instead of just showing values of data points. This makes Recharts' interactivity suited to the project's needs since NOBL Media Customers can be shown different categories within a chart and explain more of what those categories or the data point means for the customer. Due to Recharts' custom tooltip, Recharts has an edge in conveying information towards the user over the other candidates and thus, scores 3 out of 3 points in this characteristic.



Chosen Candidate

Referring to Table 3.2.7, Ant Design Charts scores the lowest (4 out of 7 points) cumulatively since its data handling capabilities are severely lacking in comparison to the other candidates. Recharts scored the second lowest (5 out of 7 points) due to its data handling. Both Ant Design Charts and Recharts could not render 100,000 data points and could render 5,000 data points in mediocre time with Recharts barely beating Ant Design Charts. ChartJS is the runner-up since it could render 100,000 data points but lacked the speed in comparison to the overall best. The leading and chosen candidate, ECharts, beat out all the other candidates due to its extremely quick data handling. Overall, ECharts is the most suitable to the project since it can be quick to render charts to the user and is highly interactive as needed for the user.

	Data Handling	Interactivity	Total
ChartJS	3/4	3/3	6/7
Ant Design Charts	1/4	3/3	4/7
ECharts	4/4	3/3	7/7
Recharts	2/4	3/3	5/7

Table 3.2.7 – Candidate Ratings

Proving Feasibility

To prove feasibility, the product must include multiple implemented graphs and charts. Due to the need to create a web application to show that other technologies can be used for other challenges as well, data visualization would need to be implemented in this product.



Section 3.3 - User Authentication

Introduction

User Authentication is the process where users log into a website to access certain information. Using a unique ID and key, technologies apply user mapping to direct clients to associate their account with subscriptions and information. After creating an account, the user can provide their ID and key to verify their identity. Login credentials are compared against original data stored in the site's server. The user is authenticated and able to access their account if the unique ID matches one created in the server.

“Every 39 seconds, a hacker strikes, contributing to the dark web's catalogue of 15 billion stolen user credentials for sale” [13]. User Authentication is a key component to ensure unauthorized users cannot gain access to sensitive information. This process is designed so that User A can open and access the information they need but cannot access any information relating to User B. Weak security for technologies that deal with this process have a much higher risk of cybercriminals hacking their system. It is important to include strong technologies that restrict the information users are authorized to access.

Ideal Characteristics

The main goal for user authentication is to allow large quantities of users to log into their respective accounts without the risk of having personal information leaked. To achieve this goal, the ideal characteristics are listed below:

Usability

This characteristic refers to ease of use. Various technologies for user authentication offer various ways to ensure the users identity, whether it is a password or something a bit different. Whatever the identification step is, it needs to be easy for the user to understand and use. NOBL Media intends for the web application to help their clients visualize their advertisement campaign statistics; they want their users to be able to log in, register for the application, and



immediately see the state of their current campaign easily. The scoring for this characteristic is rated out of 2 points with each score being described below:

1 point – A separate authentication tool with no password or key used for identification.

2 points – A username, password or some key used for identification.

Security

User authentication with weak security is at high risk for cybercriminals. Security is not only important to restrict the information users are authorized to access, but also to restrict non-clients from accessing client information. Stronger security is sometimes associated with multi factor authentication, where the user must verify their identity with a unique ID and key, and again in a second verification step. The scoring for this characteristic is rated out of 4 points with each score being described below:

1 point – One-factor authentication requiring only a username and password.

2 points – One-factor authentication requiring only a username and unique key/pin (different from an easily-guessed password).

3 points – Two-factor authentication sending interactive links for the second authentication.

4 points – Two-factor authentication with secure association to another service for the second authentication process.

Manage high amount of users

NOBL Media mostly targets companies that want to post advertisements. Since most of their clients are companies, they have hundreds of employees that may need to access information associated with the company account. This technology needs to be able to handle hundreds of users monthly. The scoring for this characteristic is rated out of 4 points with each score being described below:



1 point – Able to handle 50 users or less per month.

2 points – Able to handle 100 users or less per month.

3 points – Able to handle 1,000 users or less per month.

4 points – Able to handle 1,000 users and more per month.

Association

NOBL Media intends for its clients to mainly be companies. This characteristic is needed to make sure that companies who have many employees are all able to access their company's account and the related information. The scoring for this characteristic is rated out of 3 points with each score being described below:

1 point – Unable to have associated user-accounts.

2 points – Associated accounts achieved by one big account that many users have access to.

3 points – Associated accounts achieved by multiple user accounts associated with a company account.

Candidates

The three candidates chosen to evaluate whether they would be a good fit for the product are as follows:

Microsoft Azure

Microsoft Azure is a cloud computing service intended to build, test, deploy, and manage applications and services through data centers managed by Microsoft. This technology provides three components as a service: software, platform, and infrastructure [14]. It also supports Microsoft-specific and third-party applications that deal with different programming languages and tools.



Usability

Microsoft offers many services to many clients, because of this most of their services look similar and follow the same user interface design. Users signing into a site using their technology would see it looks like how a user signs into Outlook. After inputting a registered email and password, the authentication process within their cloud happens automatically. Since most users will recognize their log-in format, Azure scores a 2 out of 2 points for this characteristic.

Security

Azure is very popular and highly used by many companies for their websites. They have a designated security center to protect your Azure and hybrid resources. Microsoft uses a wide variety of physical, infrastructure, and operational controls to help secure Azure. Turning on the designated security center will strengthen the cloud security posture; however, it would be on the team to implement multi factor authentication or any additional actions to help safeguard the work environment. For these reasons, Azure scores a 4 out of 4 points for this characteristic.

Manage high amount of users

Azure is designed to provide an enterprise-grade cloud infrastructure on which customers and partners can rely. This includes physical elements as well as software elements like safe deployment processes, impactless maintenance, and failure prediction enabled by machine learning [14]. Built-in features that help teams design and compute critical systems across high availability, disaster recovery and backup solutions. From this, Microsoft claims that, “no matter what the service level objective may be, Azure supports your project’s reliability goals” [14]. For this reason, Azure scores a 4 out of 4 points for this characteristic.

Association

Microsoft already has services where users can share and access other information with other users. This feature is also included for user authentication where multiple users can be associated with one, bigger account. Due to their



users only being able to be associated with one big account and not with one another, Azure scores a 2 out of 3 points for this characteristic.

Google Identity

Google Sign-In simplifies integration with Google APIs through managing OAuth 2.0. On top of this, users are given the option to revoke access to an application at any time. Google Sign-In is a multi-factor authentication service where the user will verify their identity with their Gmail account [15].

Usability

Many users have already used Google Identity for various other sites. A lot of sites today even offer users to register with their Gmail account. Signing in with Google Identity looks very similar to signing into a Gmail account, and for this reason Google scores a 2 out of 2 points for this characteristic.

Security

Google Sign-In is a multi-factor authentication technology. Users are prompted to sign in with their Gmail accounts and verify their identity through Google, this is a multi-factor authentication which strengthens the security. For this reason, Google scores a 4 out of 4 points for this characteristic.

Manage high amount of users

With Google Sign-In, verification is determined by their authorized Gmail account. Google links user credentials to Gmail accounts to enable users to log in through google. This means that Google Sign-In is equipped to handle many users associated with accounts due to this google account link verification. Due to that, Google scores a 4 out of 4 points for this characteristic.

Association

Google offers multi-factor authentication by signing into a Gmail account which is associated with the registered account. The company can take this a step further with association between different accounts, not just association within the

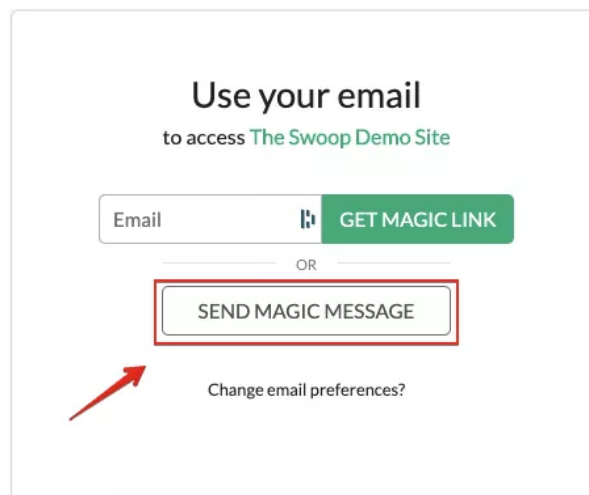
same account. With Google Identity, users will be able to access and share information with other users associated with the company, because of this Google scores a 3 out of 3 points for this characteristic

Swoop

Swoop is a simple yet secure authentication service that is password-free because they have identified passwords as highly predictable by cybercriminals. Since it is password-free, they use their patented MagicLink and MagicMessage technology to improve website security and increase customer conversion. Users will be able to sign-in one of two ways using one of the mentioned technologies. For MagicLink, the user is prompted to enter an email where an interactive link is sent, upon clicking this link the user is identified and authenticated. For MagicMessage, the user is prompted to click the "Send Magic Message" button which generates a secure email, identifying and authenticating the user once the email is sent [16].

Usability

Using those technologies, Swoop offers two ways to sign in and identify the user. The layout is easy for users to understand; they can choose either MagicLink or MagicMessage as seen on Figure 3.3.1 [16]. If the user selects MagicLink they enter an email address to which the link to authenticate is sent. However, if the user selects MagicMessage there are a few more steps required that are not listed for the user to see easily. Due to this, Swoop scored 1 out of 2 points for this characteristic.



The screenshot shows a login form with the following elements:

- Title: Use your email
- Subtitle: to access The Swoop Demo Site
- Input field: Email
- Button: GET MAGIC LINK
- Separator: OR
- Button: SEND MAGIC MESSAGE (highlighted with a red box)
- Link: Change email preferences? (pointed to by a red arrow)

Table 3.3.1 – Log-in form using Swoop



Security

Swoop offers multi factor authentication through two passwordless technologies. Since Swoop has identified passwords to be highly predictable by cybercriminals, they have included technologies that ignore this verification process. They note that since users create their own passwords, there is always a chance they will not create secure credentials. As a result, they took added stronger security measures due to weaker user verification fields. Not only do they include technologies that use multi factor authentication for added security, but they also use technologies that are passwordless to eradicate the weak, user-generated keys. For this reason, Swoop scores a 3 out of 4 points for this characteristic.

Manage high amount of users

Swoop offers three main packages to sign-up for to use Magic Link or Magic Message. Their most affordable package is the free starter package or the premium account that is free until the beginning of December 2021. The free starter package is only equipped to handle 50 monthly users while the next upgrade is only equipped to handle 1000 monthly users. Since NOBL media is trying to show all current and future clients their company's advertisement campaign statistics, this website needs to be able to handle more than 1000 monthly users [13]. Since NOBLE media would need to upgrade for more active users, Swoop scores a 2 out of 4 points for this characteristic.



Association

From Swoop's site, they do not mention anything about associated accounts. It cannot be confirmed if they offer a feature that is able to connect user accounts and for this reason Swoop scores a 1 out of 3 points for this characteristic.

Chosen Candidate

Referring to Figure 3.3.2, Google rates the highest while Swoop rates the lowest. Swoop is rated the lowest due to its inability to manage a high number of users. Azure is seemingly a decent candidate, however, this technology has too many plug-ins that are not needed for this product. Lastly, Google rates the highest since it has a familiar log-in interface for potential users, great capabilities in terms of user security, can be implemented at no cost and can manage a high amount of users. Overall, Google Identity has been chosen for user authentication.

	Usability	Security	Mange high amount of users	Association	Total
Microsoft Azure	2/2	4/4	4/4	2/3	12/13
Google Identity	2/2	4/4	4/4	3/3	13/13
Swoop	1/4	3/4	2/4	1/3	7/13

Table 3.3.2 – Candidate Ratings

Proving Feasibility

To prove feasibility, the product would need to show that users are able to sign-in using Google Identity technologies. Additionally, through this demo interface, the product would need to show that there is a two-step verification process which is deemed highly critical for information security.



Section 3.4 - API Design

Introduction

The API of the project enables NOBL Media to give customers access to its MySQL database ultimately being able to see the statistics regarding their ad campaign data. The API is what powers the web application and allows access to NOBL Media's data in a meaningful way.

Ideal Characteristics

Integration with MySQL

NOBL Media already has an established database using MySQL and therefore, the API must be able to connect and integrate with MySQL to meaningfully integrate with NOBL's existing software ecosystem. This characteristic is being rated up to 2 points with each point being described as follows:

1 point - impossible or difficult integration with MySQL

2 points - integrable with MySQL

Long Term Maintainability and Scalability

The API is what ties the web application and the NOBL Media database together and because of this, the API needs to be maintainable for NOBL Media's future usage. Additionally, the API must be able to handle a lot of data being transmitted to and from the API. Therefore, it needs to be scalable to the amount of data demand as well as the number of users using the API. This characteristic will be rated out of 3 points with each point being described as the following:

1 point - requires extensive maintenance in the future and extremely limited scalability



2 points - can be maintained in the future, but has limited scalability

3 points - maintainability is of low concern in the future due to the architecture of the API and allows for easy scalability

Security

The API is the access point through which outsiders are granted access to NOBL Media's database, which contains sensitive user information both in the personal aspect and the business aspect. Due to this, the API can be used as an entry point for attacks into NOBL Media's internal systems. Therefore, it is important that the API provides different ways to mitigate attacks into NOBL Media's systems. The rating for this characteristic is out of 3 points with each point being described as the following:

1 point - no security features added into the API architecture

2 points - security implementation is lacking or hard to implement

3 points - security implementation is understandable and there are security features within the API architecture

Speed

A product feeling fast is often a key metric for a customer deciding between different services. For NOBL Media to accomplish its mission, the API must be a responsive product equal to or better than its competitors. This characteristic is rated out of 3 with each point being described as the following:

1 point - the API design does not have any implementation that allow for quick API functionality

2 points - the API design has some implementations that allow for quick API functionality

3 points - the API design integrates speed into its architecture allowing for speedy API functionality



Candidates

There are two candidates altogether: SOAP and REST. SOAP is the older candidate in comparison to REST.

SOAP

SOAP stands for Simple Object Access Protocol and was created in 1999 as a means of enabling servers to talk to each other and create web services [21]. It was originally created by engineers at Microsoft who were unhappy with the existing Extended Markup Language Remote Procedure Call (XML-RPC) standard for APIs [21]. Microsoft officially got SOAP certified as a W3C recommendation in 2000 leading to wide adoption at the time [21].

Integration with MySQL

SOAP has limited ability to integrate with MySQL due to its rigid use of Extended Markup Language (XML) formatting. While some documentation exists for integrating SOAP XML requests into MySQL queries it always relies on bringing in a third technology to translate to and from SOAPs XML such as PHP HyperText Processor (PHP) [17]. Only Microsoft SQL server is designed to easily integrate with the SOAP design specifications. Due to SOAP's inability to fully integrate with MySQL, SOAP obtains a 1 out of 2 points for this characteristic.

Long Term Maintainability and Scalability

SOAP API's advantage in maintainability is that, compared to REST API, SOAP is strictly defined and thus unlikely to have compatibility issues due to updates to the software that uses it [18]. However, this strictly defined standard also has the problem of preventing the implementation from changing in the face of changing customer needs due to both its complexity and rigid rules. Ultimately, SOAP has an excellent advantage against REST API, however due to its incapability to react against change, it gets 2 out of 3 points in this characteristic.



Security

SOAP contains multiple built-in features designed to facilitate security. These features mainly originate from the Web Standards body of specifications which also included the original version of SOAP. Most web services that take advantage of SOAP make use of the XML Encryption, XML Signature, and SAML token to enforce data integrity and reduce data leakage from unauthorized access [18]. This suite of security tools is the most cited reason why modern APIs (especially internal ones) are built using SOAP. Due to the strong protocol features of SOAP API, it gets a 3 out of 3 points rating in this characteristic. This is easily the best selling point of SOAP API.

Speed

SOAP's full-featured nature and insistence on using XML and lack of caching adds overhead to each transaction that inevitably causes it to perform slower than the same API request using REST+JSON. For a brand-new request, this difference is small but when considering REST's ability to cache static content ahead of time the performance overhead of SOAP becomes very visible under real-world conditions [18]. Due to the limitation imposed by XML and a lack of caching, SOAP obtains a rating of 1 out of 3 points.

REST

First created in 2000 as a direct competitor to SOAP API standards, Representational State Transfer or REST, rather than being a strict protocol like SOAP, is an API architectural style designed to promote ease of implementation through a set of core principles rather than strict implementation details. It is easily the most popular style of API architecture on the modern web making up more than 70% of publicly accessible APIs [20] for this simplicity and ease of integration with query languages such as SQL and GraphQL.

Integration With MySQL

MySQL is easily the most popular relational database management system to pair with a REST API. This is typically done by having HTTP requests translated into SQL queries which are returned as a JavaScript Object Notation (JSON) file. The



specifics of how this implementation is accomplished are largely up to how the REST API is designed. Since there is an industry standard of combining REST API with MySQL servers and by extension extensive documentation [19], REST API obtains a rating of 2 out of 2 points of this characteristic.

Long Term Maintainability and Scalability

REST's increasing adoption rate among public APIs means that for the foreseeable future, REST will remain a popular API technology [20]. This means that it is likely that documentation and employee knowledge will also exist for the foreseeable future. In addition, the flexibility of REST APIs works in favor of long-term maintenance because it allows for a deviation from the original design if it obeys the design principles of REST API [19]. One example of a modification is swapping out JSON for XML. Due to the freedom in design and likely continued adoption of REST standards, REST API obtains a rating of 3 out of 3 points for this characteristic.

Security

REST makes no inherent specification for security concerns specifying only that requests are stateless and that requests are to be flagged as either cacheable or not [15]. Security measures such as authentication, rate-limiting are handled out of band by the server-side response to the HTTP traffic encapsulating the API requests. Under this limitation, security is weak unless compensated for measures external to the API itself. Since security measures will be implemented to compensate due to the lack of security features of REST API, REST obtains a rating of 1 out of 3 points.

Speed

One of the biggest reasons why REST has such a competitive lead on SOAP in terms of adoption numbers is it is a considerably faster API architecture for several reasons. The biggest reason is that, unlike SOAP, REST APIs are free to use simple formats with little overhead such as JSON. The other is that REST mandates that responses should be cacheable which speeds up response times especially in situations with a high round trip time for a connection [20]. Speed and simplicity are the defining characteristics of why REST API is a popular choice for publicly available APIs and therefore, REST API obtains 3 out of 3 points.



Chosen Candidate

Referring to Table 3.4.1, the chosen candidate is REST API since it scored 3 points higher than SOAP API. This is due to REST API's long-term maintainability and scalability as well as its speed. SOAP API lacked the speed and most importantly, could not integrate well with MySQL, the client's choice for database. Overall, a REST API is the ideal solution for this challenge.

	Integration with MySQL	Long Term Maintainability and Scalability	Security	Speed	Total
SOAP API	1/2	2/3	3/3	1/3	7/11
REST API	2/2	3/3	1/3	3/3	9/11

Table 3.4.1 – Candidate Ratings

Proving Feasibility

In the future, to prove feasibility, we would need to show a REST API working to retrieve data specifically from NOBL Media's database and directly sending it to a prototype of the web application.

Referring to Figure 3.4.2, the making of a request to a demo REST API which makes a SQL query in a database of employee data at a fictional company can be seen.



```
"profile_image": ""
},
{
  "id": 22,
  "employee_name": "Yuri Berry",
  "employee_salary": 675000,
  "employee_age": 40,
  "profile_image": ""
},
{
  "id": 23,
  "employee_name": "Caesar Vance",
  "employee_salary": 106450,
  "employee_age": 21,
  "profile_image": ""
},
{
  "id": 24,
  "employee_name": "Doris Wilder",
  "employee_salary": 85600,
  "employee_age": 23,
  "profile_image": ""
}
],
"message": "Successfully! All records has been fetched."
}
→ ~ curl https://dummy.restapiexample.com/api/v1/employees | jq | tee out.txt
```

Figure 3.4.2 – REST API Demo

While a demo of the same goal like in the REST API demo was harder to implement, Figure 3.4.3 is a SOAP API demo that works in a very similar way save for using a POST request containing the SOAP XML structure.

```
import requests
req_headers = {"content-type": "text/xml"}
req_body = "<?xml version='1.0'?">"
req_body += "<soap:Envelope xmlns:soap='http://www.w3.org/2003/05/soap-envelope'"
req_body += "<soap:Header></soap:Header>"
req_body += "<soap:Body>"
req_body += "<m:GetUser>"
req_body += "<m:UserId>123</m:UserId>"
req_body += "</m:GetUser>"
req_body += "</soap:Body>"
req_body += "</soap:Envelope>"
response = requests.post(
    "http://www.example.com/exampleapi",
    data=req_body,
    headers=req_headers
)
soap.py (END)
```

Figure 3.4.3 – SOAP API Demo



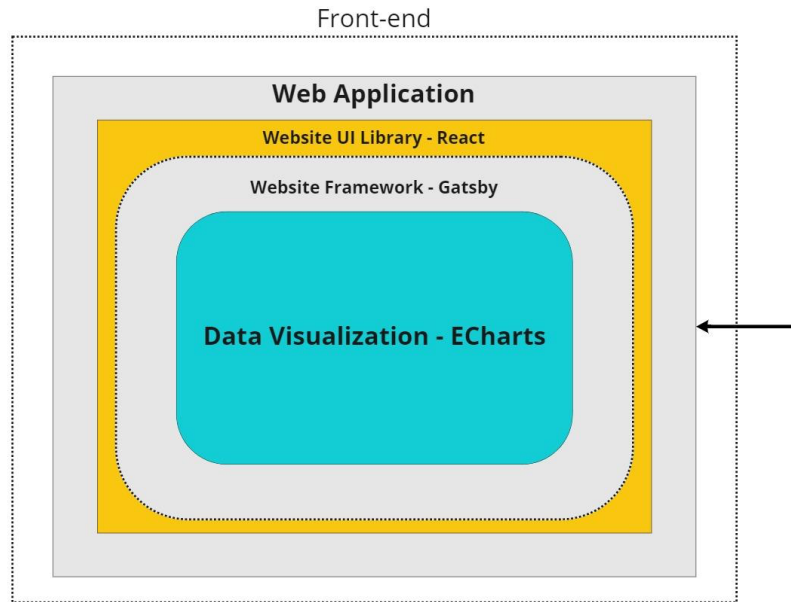
Chapter 4 - Technological Integration

The project requires the ability to effectively integrate all the chosen software components into one cohesive system. One mitigating factor of this challenge is that only certain pieces need to interact with each other. For example, MySQL only needs to be compatible with the chosen API rather than the chosen rendering toolkit because the API acts as an intermediary between the two pieces of software.

The chosen data visualization library, ECharts, has built-in methods for parsing JSON data explicitly to handle REST API responses, which in turn are well suited to making requests to MySQL database servers. Additionally, the data visualization library has been checked to ensure solid compatibility with the site rendering toolkit and the use of TypeScript instead of plain JavaScript.

While incompatibility specific to this project may arise, this toolchain is an industry-standard and well vetted for compatibility and suitability for this client's typical use case. We are confident that technological compatibility issues will not be a significant issue for this project.

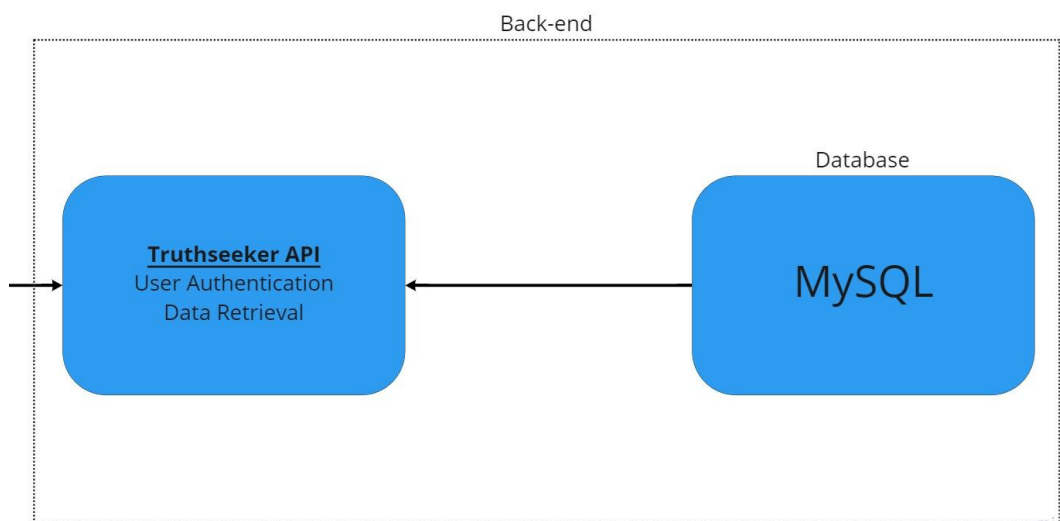
Referring to Figure 4.1, the figure shows the front-end portion of the project which consists solely of the web application. Within the web application is where we will be using a website UI library, React, and the website framework, Gatsby. These two technologies combined will allow this web application to provide the user with quick loading times and smooth interactivity. Another component of the web application is the data visualization which allows the abstraction of the data of the user's ad campaign. The website UI library, the website framework, and the data visualization library combined make up the web application. To connect the web application to NOBL Media's database, which stores all ad campaign data, the project also requires an API which allows the web application to "communicate" with NOBL Media's database.



miro

Table 4.1 – Project Front-End System Diagram

The API's main job is to retrieve data from the database and send it to the web application. However, it is also the API's job to make sure that users can create an account, be able to authenticate themselves when logging in, and then storing all user information back to the database. This API will be a REST API built with Node.js and Express. Referring to Figure 4.2, the back-end of the project consists of NOBL Media's database, which uses MySQL as its database service, and the API. Together they are shown in Figure 4.3 which shows the entire system of the project.



miro

Figure 4.2 – Project Back-End System Diagram

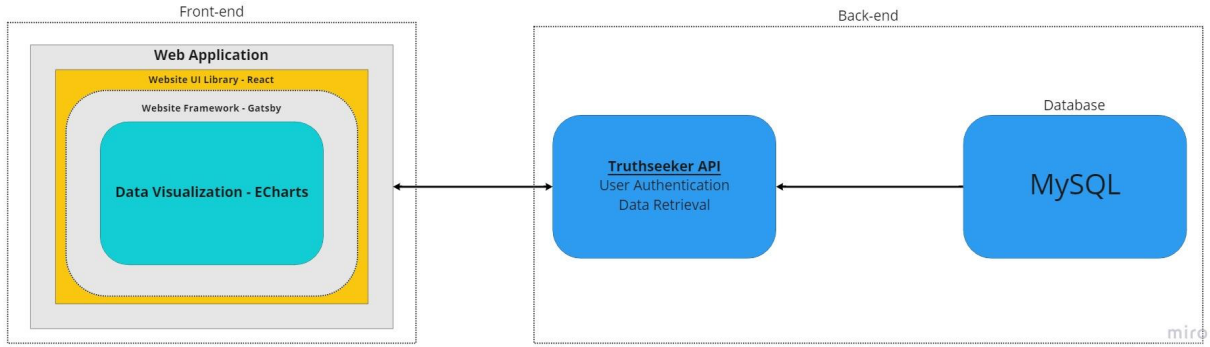


Figure 4.3 – Project System Diagram



Chapter 5 - Conclusion

Misinformation has never been more powerful than it is today. Relying on people to discern truth from mistruth has led us to a society where some of us live in different realities. The reason why misinformation is so prevalent today is because there is money to be made by creating falsehoods served with ads on the internet. These web pages with misinformation must have ads removed from them to defund the people creating these lies. NOBL Media has created an artificial intelligence with a database for the Misinformation and Credible News Analysis Tool. Team Truthseeker is tasked with completing NOBL Media's business flow by developing an API and website dashboard.

Team Truthseeker will develop the web application UI with React and Gatsby written in TypeScript. React was the preference of our client while Gatsby's speed and client-side generation made it a natural choice to pair with React. ECharts was chosen because it is the most powerful data representation tool out of all the candidates that were compatible with React and Gatsby. ECharts' direct process of creating graphs and charts allows for understandability of the project's data visualization process. Google Identify will be handling the user authentication due to its familiarity and reliable security. The final piece of our stack, a REST API, was chosen based on its speed and support with NOBL Media's MySQL server.

Now that the project has its chosen candidates together, further development of the Credibility and Misinformation News Analysis Tool can begin. NOBL Media has provided Team Truthseeker with an Amazon Web Services (AWS) server environment to begin developing a prototype. In the next three weeks, Team Truthseeker will develop a prototype dashboard website with a working API allowing a coherent system to be realized.



Chapter 6 - References

- [1] "Cigarettes were once 'physician' tested, approved," *Healio*, 10-Mar-2009. [Online]. Available: <https://www.healio.com/news/hematology-oncology/20120325/cigarettes-were-once-physician-tested-approved>. [Accessed: 29-Oct-2021].
- [2] *The Health Consequences of Smoking—50 Years of Progress: A Report of the Surgeon General*. Centers for Disease Control and Prevention (US), 2014. [E-book] Available: <https://www.ncbi.nlm.nih.gov/books/NBK179276/>
- [3] J. Bailly, "Misinformation and Credible News Analysis Tool." NOBL Media, 2021. [Online] Available: https://www.ceias.nau.edu/~edo/Classes/CS_Capstone/Projects/F21/Bailly-NOBL.pdf
- [4] *Dynamic typing vs. static typing*, Mar-2015. [Online]. Available: https://docs.oracle.com/cd/E57471_01/bigData.100/extensions_bdd/src/cext_transform_typing.html. [Accessed: 31-Oct-2021].
- [5] "The history of react.js on a timeline," *The History of React.js on a Timeline*, 11-Oct-2021. [Online]. Available: <https://blog.risingstack.com/the-history-of-react-js-on-a-timeline/>. [Accessed: 31-Oct-2021].
- [6] "Fastest Static-site generation web framework," *Gatsby*. [Online]. Available: <https://www.gatsbyjs.com/>. [Accessed: 29-Oct-2021].
- [7] "Start building amazing web experiences," *Fastest Static-Site Generation Web Framework*. [Online]. Available: <https://www.gatsbyjs.com/why-gatsby/>. [Accessed: 19-Oct-2021].
- [8] "Next.js by vercel - the REACT framework," by Vercel - *The React Framework*. [Online]. Available: <https://nextjs.org/>. [Accessed: 29-Oct-2021].
- [9] Y. Adhikary, "Client side rendering vs server side rendering in REACT JS & next JS," *Medium*, 15-Dec-2020. [Online]. Available: <https://yudhajitadhikary.medium.com/client-side-rendering-vs-server-side-rendering-in-react-js-next-js-b74b909c7c51>. [Accessed: 29-Oct-2021].
- [10] "Echarts-for-react," *npm*. [Online]. Available: <https://www.npmjs.com/package/echarts-for-react>. [Accessed: 03-Nov-2021].
- [11] "React-chartjs-2," *npm*. [Online]. Available: <https://www.npmjs.com/package/react-chartjs-2>. [Accessed: 29-Oct-2021].



- [12] *Recharts*. [Online]. Available: <https://recharts.org/>. [Accessed: 29-Oct-2021].
- [13] B. Wallace, "What is Authentication & Why is it important?," *Dumb Little Man*, 21-Sep-2020. [Online]. Available: <https://www.dumblittleman.com/why-is-authentication-important/>. [Accessed: 28-Oct-2021].
- [14] "Create your azure free account Today," *Microsoft Azure*, 2021. [Online]. Available: https://azure.microsoft.com/en-us/free/search/?OCID=AID2200277_SEM_acd3eaf92b3c1e4e38b08b867e65a73d%3AG%3As&ef_id=acd3eaf92b3c1e4e38b08b867e65a73d%3AG%3As&msclkid=acd3eaf92b3c1e4e38b08b867e65a73d. [Accessed: 01-Nov-2021].
- [15] "Integrating google sign-in Into your web app," *Google Identity*, 14-Sep-2020. [Online]. Available: <https://developers.google.com/identity/sign-in/web/sign-in>. [Accessed: 25-Oct-2021].
- [16] "User authentication: Understanding the basics & top tips," *Swoop Password-Free Authentication*, 20-Jul-2020. [Online]. Available: <https://swoopnow.com/user-authentication/>. [Accessed: 26-Oct-2021].
- [17] Alam, Masud. "Soap with PHP and Mysql." *w3programmers*, 26 March. 2013, <https://www.w3programmers.com/soap-with-php-and-mysql/>.
- [18] Altvater, Alexandra. "Soap vs. REST: The Differences and Benefits between the Two Widely-Used Web Service Communication Protocols." *Stackify*, 30 March 2021, <https://stackify.com/soap-vs-rest/>.
- [19] Fitzgerald, Anna. "Soap vs REST Apis: The Key Differences Explained for Beginners." *HubSpot Blog*, 29 September 2021, <https://blog.hubspot.com/website/rest-vs-soap>.
- [20] Brown, David. "Quickly Build a REST API for a Mysql Database." *Exemacro*, David, 23 September 2020, <https://exemacro.com/quickly-build-a-rest-api-for-a-mysql-database/>.
- [21] "Use soap API in web applications - SQL server reporting services (SSRS)," *Use SOAP API in Web Applications - SQL Server Reporting Services (SSRS) | Microsoft Docs*. [Online]. Available: <https://docs.microsoft.com/en-us/sql/reporting-services/application-integration/integrating-reporting-services-using-soap-web-application?view=sql-server-ver15>. [Accessed: 01-Nov-2021].