Software Requirements Specification

for

Comorbidity and Genetic Factors and their Impacts on Patients with COVID-19

Version 1.1.3 approved

Prepared by

Rohan Chatterjee	Project Lead
Riese Atianzar	User Interface Designer
Francisco Contreras	User Experience (UX) Engineer
Luis Gonzalez	User Experience (UX) Engineer
Emily Gonzalez	Data Visualization Engineer
Ting Fung Ha	Technical Writer
Juan Hernandez	Data Analyst
Carlos Hernandez	Data Gatherer/Developer
Jimuel Cedrick Julaton	Data Visualization Engineer
Chen-Ching Lin	Data Visualization Engineer
Dr. Navid Amini	Faculty Advisor
Haley Kirk	Project Liaison

Sponsored by Vodafone

21 November 2022

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Ting Fung Ha	November 21, 2022	Document creation	1.0.0
Ting Fung Ha	November 26, 2022	First Draft Completed	1.0.1
All members	November 28, 2022	First Group Revision	1.0.2
All members	December 1, 2022	Second Revision	1.0.3
Francisco Contreras	December 7, 2022	Revision/additions to document	1.0.4
Francisco Contreras	December 8, 2022	Revision/Additions to document	1.0.5
Ting Fung Ha	March 11, 2023	First Draft of Clinical Analysis	1.1.0
Ting Fung Ha	March 15, 2023	First Revision on Clinical Analysis	1.1.1
All members	April 4, 2023	Revisions/Additions to document	1.1.2
All members	May 1, 2023 - May 4, 2023	Group Revisions, Feedback & Comments	1.1.3

1. Introduction

1.1 Purpose

This document will provide a detailed explanation of a COVID-19 data visualization application. It will cover the application's components, purpose, data flow, and maintenance methods. In addition, the document will discuss the utilization of machine learning principles and a feature selection algorithm to investigate the correlation between blood types and COVID-19 mortality and severity.

In addition, the document will describe the tools and technologies used to develop the application, along with how they were implemented. Finally, the document will provide a brief explanation of the application's functionality and how the aforementioned technologies were utilized to achieve accurate results.

1.2 Intended Audience and Reading Suggestions

Analysts: It is recommended for analysts to go through Section 2 to 3 and 5 to 7. These sections will introduce some of the origin of the datasets, along with tools and technologies applied to this project.

Data Scientists: Data scientists should also read Section 2 to 7 of this document.

Medical Professionals: Medical professionals can refer to Section 2 and 5 for more relevant information.

For anyone with an interest in the project, it is recommended to read through the entire document to get a full understanding of this project.

1.3 Product Scope

The application should run flawlessly on a wide variety of devices, including PCs, smart devices running on Android and iOS platforms. However, an internet connection is required to access all of the latest contents.

Due to the confidentiality of the datasets used in the machine learning portion, the team is not permitted to share the data with any individuals outside of the group. As a result, readers and users will only be given a description of the data and a summary of the results.

1.4 Definitions, Acronyms, and Abbreviations

Refer to Appendix A.

1.5 References

1. Data-Driven Document (D3) Overview

   https://d3js.org/

2. Project Jupyter's Origins and Governance

   https://jupyter.org/about

3. Node.js

   https://nodejs.org/

4. GitHub

   https://github.com/

5. Material UI

   https://v4.mui.com/

6. Tittle: Software Requirements Document (Fall 2021)

   Authors: Antonio Campos, Amy Guttman, Alec Kaczmarek, Vincent Li, Saiyang Liu, Ricardo Marroquin, Miguel Nonoal-Garcia, Alexandra Strong, Jonathan Sum, Edwin Zapata Minero

   Date: November 27, 2022

7. Title: A Mobile Assistive Technology for Peripheral Visual Field Loss

   Authors: Daniel Esparza, Canhong Huang, Jonathan Kan, Abran Lezama Pastor, Brenden Mccabe, Ashley Munoz, Uchenna Onuigbo, Duy Pham, Nicolas Sandoval, Jacob Schultz

   Date: November 27, 2022

2. Overall Description

2.1 System Analysis

Countless lives were changed due to the COVID-19 pandemic. The world had to quickly evolve as cases increased and new variants began to emerge. As lock down began, COVID-19 containment and vaccination became a priority for governments and the medical industry. Millions lost their lives, while many more recovered and move on to a post-COVID-19 world. This project is focused on a possible attributes that might have been overlooked by the professional industry. This is done through two parts. First is the creation of an application with visualizations that show relevant COVID-19 factors and release it for the public to see.And the second component is, a clinical analysis using machine learning is used to determine whether different blood types can affect mortality and severity of the virus.

Part I: Visualizations

The first part of the project allowed all the members in the group to learn, research and analyze the data obtained from trustful organizations. Next, the team created visualizations to identify the resemblances between some of the key attributes.

The application is created to describe the significance of the virus with interactive visual illustrations. Users with access to the application should be able to customize their own view windows by adjusting some of the attributes provided from datasets posted from different trusted organizations.

Some of the major obstacles for this project are:

- Learning the different types of tools and how to implement it
- Finding a method to deal with the large quantity of data

These obstacles were crossed with instructional sourser, guidelines and documentations from the respective tools and demonstrations from other educational websites.

Part II: Clinical Analysis with Machine Learning

Although displaying the graphical representations about the COVID-19 virus to the public is important, it is equally important to study patients with COVID-19 and determine if overlooked comorbidity and genetic factors play a factor into them getting infected. In order to

achieve such goals, the concept of machine learning and a feature selection algorithm was implemented by the team. The algorithm will first create a duplicate dataset based on the original set, but with modified values (the duplicate dataset is known as the Shadow Features). Next, in the Binomial Distribution Stage, the shadow features will be compared with the original dataset to see if the comparison values are random or not. If the effects are randomized, it means the shadow features do not provide enough information to make itself "important". On the other hand, if the shadow features returned constant importance, then the shadow features are deemed to be "more important". After a number of computations, the algorithm will be able to determine the shadow features with the "most importance".

Some of the obstacles in the second half we encountered are, but not limited to:

- The algorithm can be difficult to get accurate and desired data.
- The algorithm is fairly complex to operate due to multiple computations required

2.2 Product Perspective

Our application shares some functionalities from the COVID-19 trackers created by John Hopkins University. Both applications take in data from trusted sites and convert it into visualizations for the public to see. However, our application will focus more on the visualization aspect rather than just simply tracking it. The customization view window is based on the user's preference and is completely independent from the tracker. Therefore, the visualizations will allow users to interact and generate customized views based on the selected dataset. Aside from that, our application will allow the users to capture and store some of the visualizations into a built-in dashboard for the users to visit at another time.

The clinical analysis with machine learning shares a number of similarities with the other analysis in the scientific community, mainly on the selection algorithm and the machine learning models.

2.3 Product Functions (w/o Boruta)

2.3.1 Home Screen

This is the first page every user will see during their visit. The user will be able to select a dataset based on domestic, or international.

2.3.2 Navigation Bar

The navigation bar will act as a shortcut for users to access some of the useful pages quickly.

2.3.3 Data Visualization Page

After a dataset is selected from the home screen, the user will be transferred to a page with all its relevant visualizations. The visualization in the center will have functions like filter and dashboard for later usage.

2.3.4 Data Filter

Once a visualization is selected in the Data Visualization Page, the user will be able to see the dataset with some boundaries placed by the user such as date, color, state, etc.

2.3.5 Your Dashboard

The dashboard will display any visualizations when the user clicks on the “save to dashboard” in the Data Visualization Page. This function will allow the user to compare visualizations side-by-side, or use them at another time.

2.4 User Classes and Characteristics

The application does not have any user classes implemented. All users will see and interact based on the same access level.

2.5 Operating Environment

Due to the excellent compatibility of JavaScript and its libraries, this application should not create problems in any of the Chromium based, iOS or Microsoft browsers. Any mobile users with an Android (version 10 or above) or iOS (version 14 or above) based system should have full access to this application without any hassles. Finally, the Tableau Portal will run on any browsers with an internet connection.

2.6 Design and Implementation Constraints

Developers will need to know the following tools before starting:

- D3.js
- Dataset Providers
- GitHub
- JavaScript
- Material UI
- Node.js
- Python 3.7+
- React

2.7 Assumptions and Dependencies

Any user who wishes to access the application is expected to have a device with an internet connection. The device is recommended to have an up-to-date modern browser and operating system. We plan to use the libraries React, D3, and Material UI to assist us in making an efficient application.. For any detailed requirements, please refer to 2.5 and 2.6.

The clinical analysis was implemented with Python 3 and the following libraries:

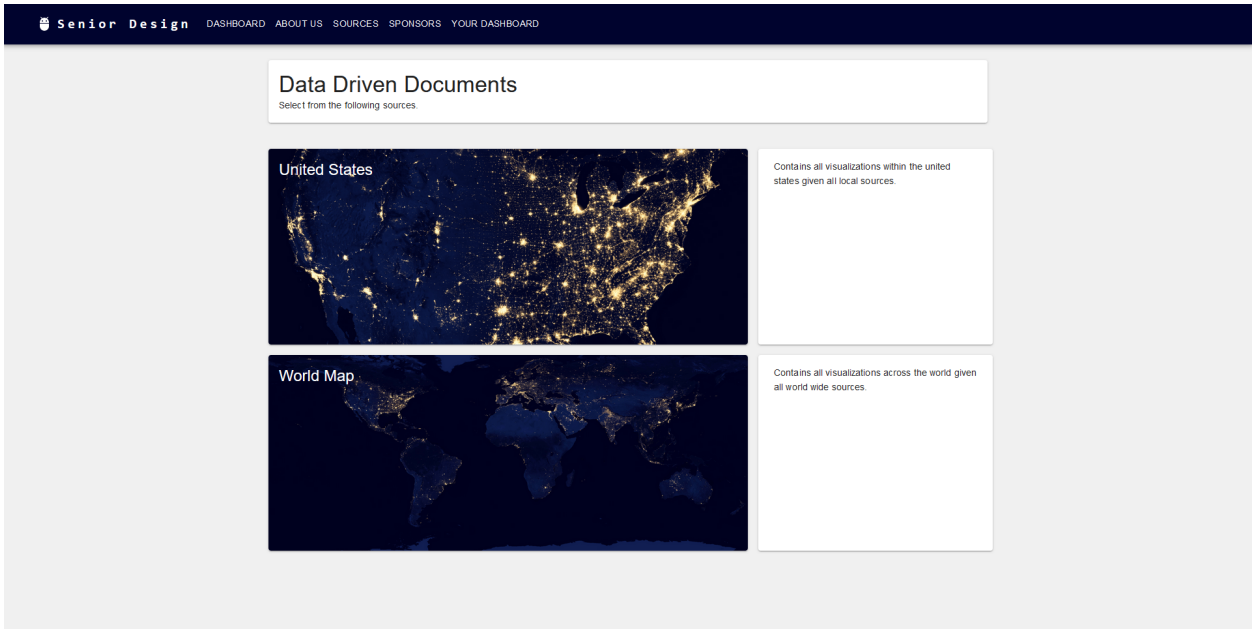
- Numpy
- Python Analysis Data Library (PANDAS)
- Matplotlib
- SciKit-Learn (SKLearn)
- Boruta Algorithm implementations

2.8 Apportioning of Requirements

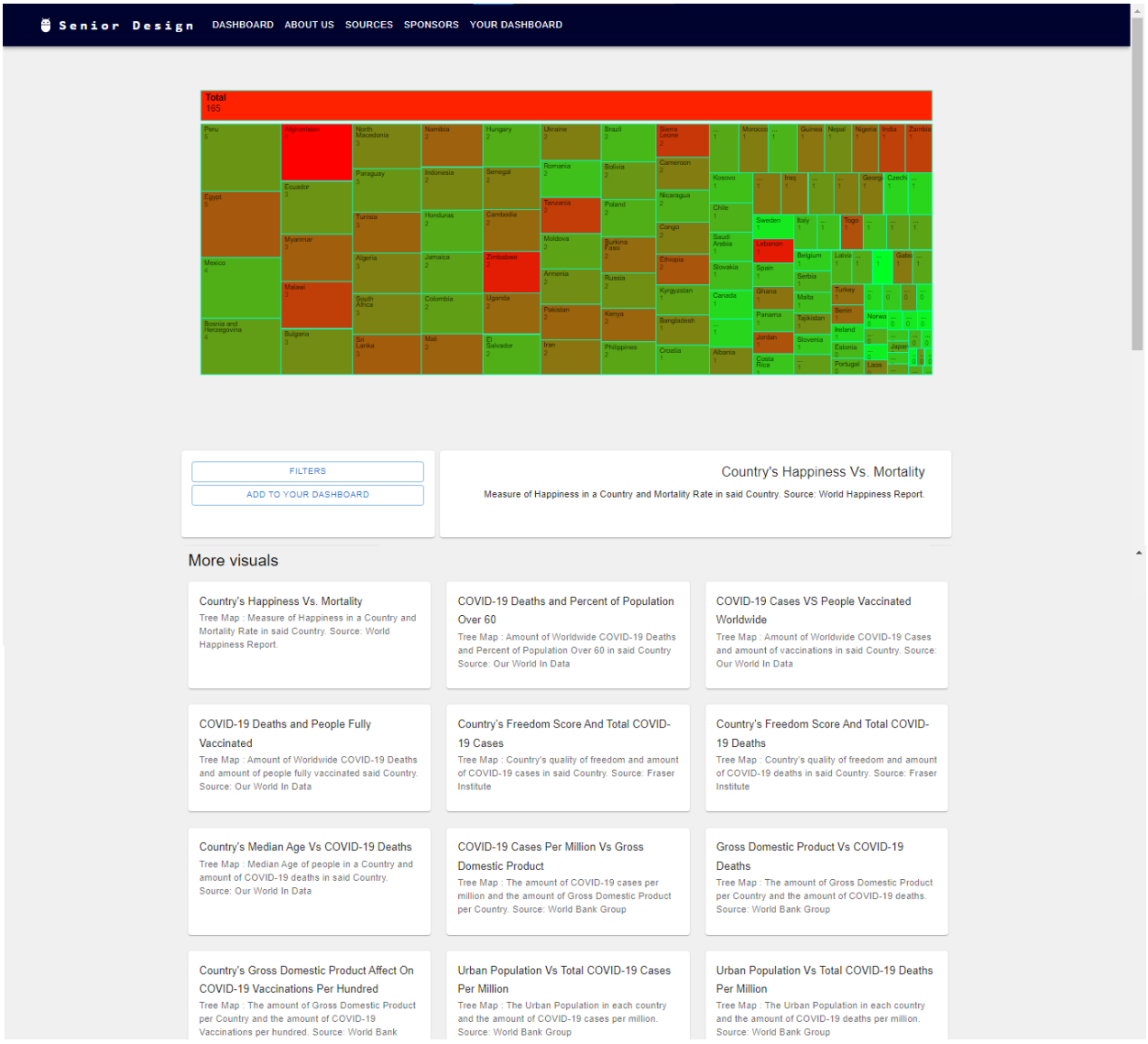
The amount of visualizations and the size of the dataset will cause the application to take longer to load its contents. On top of that, the tools will have to utilize more computation power to convert these datasets into visualizations. On the other hand, the various filters will slow down the performance of the application. The convenience allows the user to view the dataset based on their own preference. The drawback, however, will take longer to generate the view.

3. External Interface Requirements

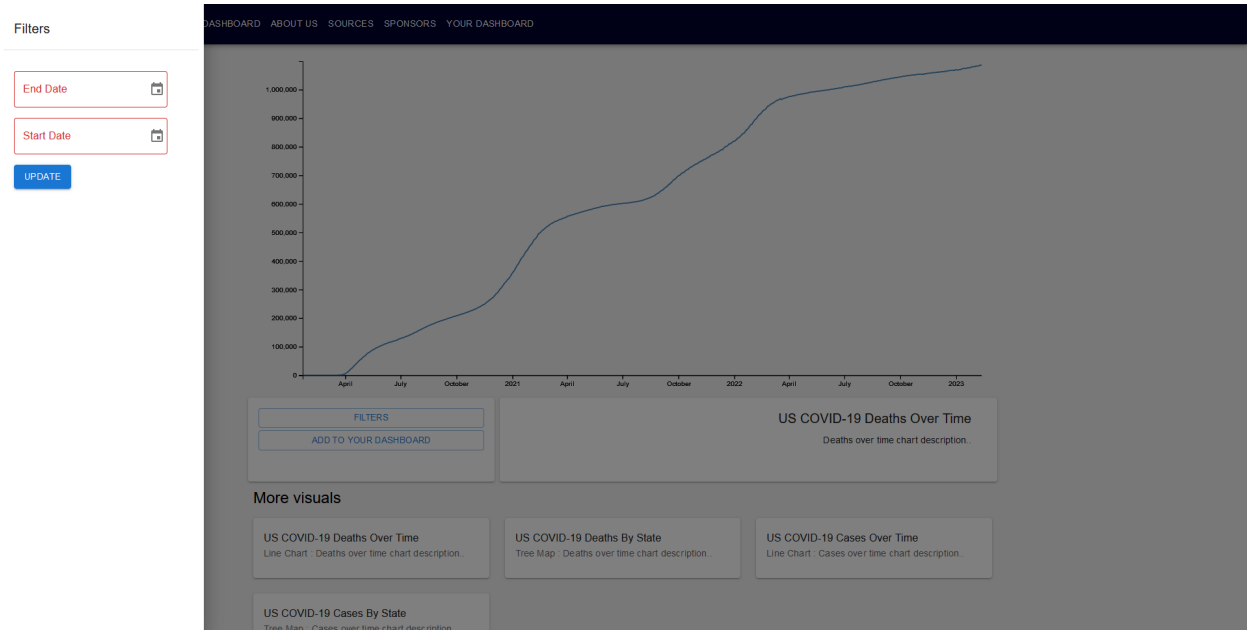
3.1 User Interfaces



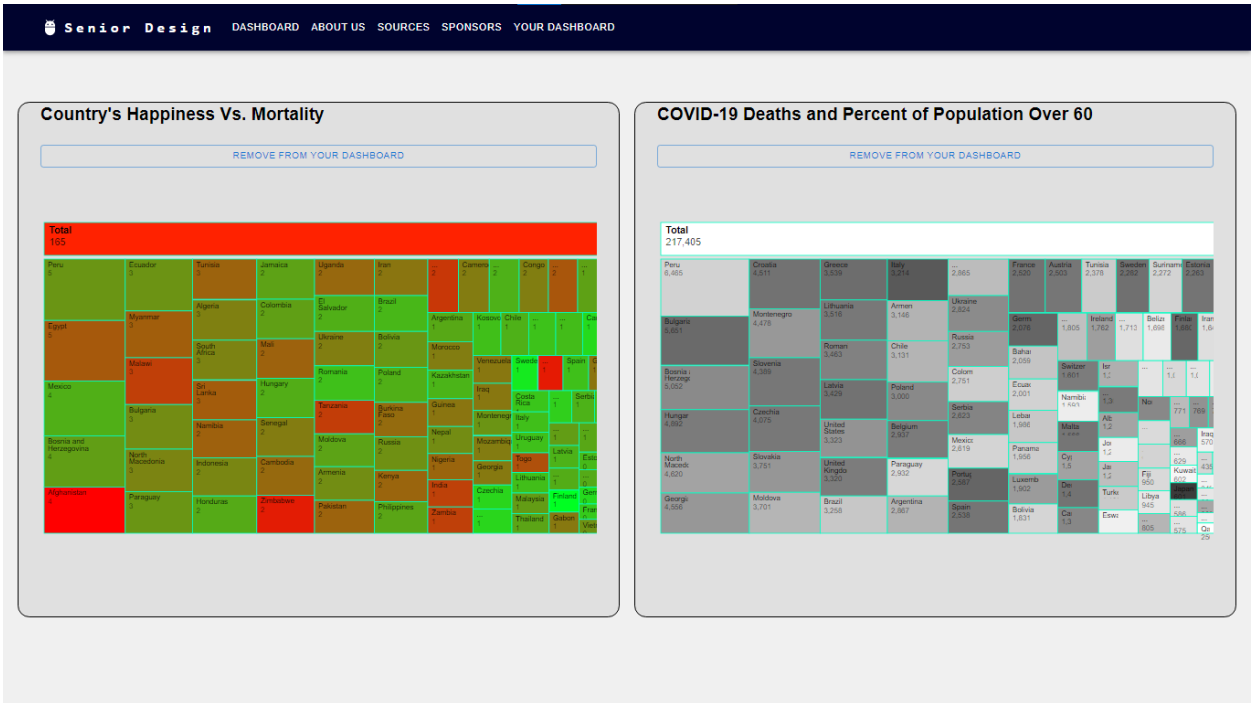
This is the homepage of our application. All users will be greeted by this screen with two types of datasets, domestic and international. The pictures and descriptions are all placed in separate card functions to represent an organized and focused view.



Once a dataset is selected, the user will be directed to another page with all the relevant visualizations. The visualization in the center can be customized based on some of the filter options, or get added to the dashboard for side-by-side comparison with other visualizations, or use them at a different time.



This sidebar interface will be the interface that will appear once the user clicks the filter button. This interface will allow the user to filter to see the visualizations they desire. The sidebar will disappear once the user clicks on somewhere on the application that is not the sidebar.



This is the “Your Dashboard” page, a second function implemented that allow the user to save the visualization to another page for side-by-side comparison, or group a collection of visualizations together for future usage.

3.2 Hardware Interfaces

There are no hardware interfaces available at this time.

3.3 Software Interfaces

The project utilized various software programs to achieve our goals. In the first part of the project, the team used Tableau Public and Tableau Desktop for data analysis and created visualizations. The results were published using Tableau Public and are accessible as a website.

A number of JavaScript libraries and APIs were selected during the development phase. These tools include:

- D3.js
- JavaScript Fetch API
- Material UI

- Node.js
- React
- Python

3.4 Communications Interfaces

The application can be accessed with a valid URL. This is achieved with a browser, the JavaScript Fetch API, D3 API, HTTP connection and a stable internet connection. The encryption will be handled by the browser, while data transfer rate will depend on the internet connection.

4. Requirements Specification

4.1 Functional Requirements

The application shall read through an external dataset to create interactive visualizations

The application shall use a filter system that displays visualizations based on the user's preference.

The application shall display previews of visualizations or full-sized visualizations from the desired dataset.

The application shall change interfaces depending on the user's input.

The application shall keep track of most up-to-date entries from used datasets.

The application shall provide another page for users to view all of the saved visualizations based on user input.

The application shall retain all of its functionalities for users with different device platforms.

4.2 External Interface Requirements

There are no applicable requirements for this software at this time

4.3 Logical Database Requirements

The application will not use a database to generate visualizations, but APIs were used to fetch for additional resources.

No databases were used at this time.

4.4 Design Constraints

Due to the large file sizes, the team will have to come up with a procedure to handle the files in a short amount of time, while generating impactful visualizations. The team will also have to think of a method to display the same contents with an optimized viewing windows for different devices.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The performance of this application will depend on the following:

- Device hardware specification
- Internet quality
- Browser and its version
- Future updates for JavaScript and its relevant libraries

Devices with access to the Internet, able to load a web page with images should be able to see all the contents within a short period of time.

5.2 Safety Requirements

No safety requirement specified.

5.3 Security Requirements

Given that real world sensitive medical data have been applied during our clinical analysis phase, each team member is required to maintain the integrity of all data. At the same time, each member must protect and keep the data confidential.

5.4 Software Quality Attributes

The software will be compatible with different kinds of devices across Android and iOS, personal computers, tablets, etc. The datasets will be monitored regularly to ensure its accuracy and trustworthiness.

6. Legal and Ethical Considerations

There are some legal and ethical considerations, mainly from the datasets and its usages. Each member from the team is required to keep the data safe and accurate without any form of manipulation. At the same time, every member cannot defame the dataset providers for any reason.

Additional considerations and restrictions will apply to the team in clinical analysis, where each member must keep the data confidential and protect its integrity.

Appendix A: Glossary

API: Application Programming Interface

GUI: Graphical User Interface

iOS: An operating system installed on all Apple mobile devices

JS: JavaScript

React: A JavaScript library specifically designed for user interface

Material UI: A language based on Material Design, created by Google.

D3.js: A JavaScript library for manipulating HTML data

Node.js: A cross-platform JavaScript runtime environment

Appendix B: To Be Determined List

Some of the functions that may be implemented in the future includes:

- Error messages and functions for overflow handling
- Provide more attribute customizations for a dataset (e,g, COVID-19 variants and etc.)
- More customization windows based on their attributes (Create visuals if the dataset contains statistics for COVID-19 variants etc.)