Software Design Document

for

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

Version 1.1.4 approved

 Prepared by

 Rohan Chatterjee Project Lead

 Riese Atianzar Developer

Francisco Contreras Developer

Luis Gonzalez Developer

Emily Gonzalez Developer

Ting Fung Ha Developer

Juan Hernandez Developer

Carlos Hernandez Developer

Jimuel Cedrick Julaton Developer

Chen-Ching Lin Developer

Dr. Navid Amini Faculty Advisor

Sponsored by Vodafone

21 November 2022

**Table of Contents**

Table of Contents................................................................................................................. pg 3

Revision History................................................................................................................... pg 5

 1. Introduction................................................................................................................ pg 6

 1.1. Purpose........................................................................................................... pg #

 1.2. Intended Audience and Reading Suggestions................................................ pg #

 1.3. Product Scope................................................................................................ pg #

 1.4. Definitions, Acronyms, and Abbreviations .................................................. pg #

 1.5. References...................................................................................................... pg #

 2. Overall Description.................................................................................................... pg #

 2.1. System Analysis…......................................................................................... pg #

 2.2. Product Perspective........................................................................................... pg #

 2.3. Product Functions........................................................................................... pg #

 2.4. User Classes and Characteristics.................................................................... pg #

 2.5. Operating Environment.................................................................................. pg #

 2.6. Design and Implementation Constraints........................................................ pg #

 2.7. User Documentation...................................................................................... pg #

 2.8. Assumptions and Dependencies.................................................................... pg #

 2.9. Apportioning of Requirements...................................................................... pg #

 3. External Interface Requirements............................................................................... pg #

 3.1. User Interfaces............................................................................................... pg #

 3.2. Hardware Interfaces....................................................................................... pg #

 3.3. Software Interfaces........................................................................................ pg #

 3.4. Communications Interfaces........................................................................... pg #

 4. Requirements Specification....................................................................................... pg #

 4.1. Functional Requirements............................................................................... pg #

 4.2. External Interface Requirements................................................................... pg #

 4.3. Logical Database Requirements.................................................................... pg #

 4.4. Design Constraints......................................................................................... pg #

 5. Other Nonfunctional Requirements........................................................................... pg #

 5.1. Performance Requirements............................................................................ pg #

 5.2. Safety Requirements...................................................................................... pg #

 5.3. Security Requirements................................................................................... pg #

 5.4. Software Quality Attributes........................................................................... pg #

 5.5. Business Rules............................................................................................... pg #

 6. Legal and Ethical Considerations.….......................................................................... pg #

Appendix A: Glossary........................................................................................................ pg #

Appendix B: Analysis Models........................................................................................... pg #

Appendix C: To Be Determined List................................................................................. pg #

Revision History

| Name | Date | Reason For Changes | Version |
| --- | --- | --- | --- |
| Ting Fung Ha | 21 November 2022 | Document created | 1.1.1 |
| Ting Fung Ha | 27 November 2022 | First draft completed | 1.1.2 |
| All Members | 28 November 2022 | First Group Revision | 1.1.3 |
| Rohan Chatterjee,Ting Fung Ha | 1 December 2022 | Revision with Project Lead | 1.1.4 |

**1. Introduction**

**1.1 Purpose**

The document will provide the users with a general idea of the project, which is split into two parts. Each part will include the components involved, its functionalities, and how the components work together. The document will also cover some of the design strategies, concerns, challenges and various restrictions before using each portion of the project.

**1.2 Document Conventions**

 The document is written in a minimalistic and a straightforward style. Therefore, no typical conventions were followed.

**1.3 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.

**1.4 System Overview**

The application will create visualizations based on the data obtained from trusted organizations. After that, users will have the option to focus on one of the visualizations in depth and see its relationship with other categories from the same dataset..

**2. Design Considerations**

**2.1 Assumptions and Dependencies**

The first part of the project mainly covered the experiences our team had with Tableau products (Tableau Public, Desktop and Portal). These products can be installed on computers with the following specifications:

Windows OS: 8 / 8.1 / 10(x64)

Macintosh OS: Mojave 10.14 / Catalina 10.15 / Big Sur 11.4+ / Monterey 12.6+ (for Tableau 2022.3+)

The second part of the project is an application that can generate multiple visualizations based on a dataset. To achieve this, a number of JavaScript libraries were used. Please refer to Section 5, subsection 1 for a list of tools and technologies used.

**2.2 General Constraints**

Describe any global limitations or constraints that have a significant impact on the design of the system's software (and describe the associated impact). Such constraints may be imposed by any of the following (the list is not exhaustive):

· Given the majority of the project is based on software, there will be very little hardware limitations. However, the device should have enough hardware computing power to stream online contents with short delays.

· In order to access this application, view the latest datasets and their visualizations, users must have a stable internet connection to access all of its functions.

· The visualizations rely heavily on accurate datasets. Therefore, all the datasets came from trusted sites in the public and private sectors.

**2.3 Goals and Guidelines**

The main goal of the application is to get a better understanding of the COVID-19 virus. The team wants to learn and analyze the virus in various unique aspects which none has ever considered before. Some of these aspects can involve private patient information like blood type, weight and others. In the end, the product has to be delivered by December 2nd, 2022.

**2.4 Development Methods**

The overall development principles for this project would be a form of Unplanned Mad Scramble Development. However, trial and error was also applied in the first half of this project. This happened during the visualization creation stage, where we had to use the categories with the highest significance in the dataset.

**3. Architectural Strategies**

**3.1 Product Usage**

**3.1.1 Part 1 – Tableau Portal**

3.1.1.1 Tableau Public: Tableau Public was used due to its functionality of rendering different types of visualizations with a dataset. The abundant customization options provided team with various new insights, which aided us in the decision making process.

3.1.1.2 Tableau Desktop: The team decided to use Tableau Desktop because it is an upgrade of Tableau Portal with additional enhanced abilities. A number of the datasets were so large in file size where Tableau Public cannot handle it. As a result, Tableau Desktop was chosen as a replacement for Tableau Public.

 **3.1.2 Part 2 – An Application with Visualizations from Datasets**

3.1.2.1 D3.js: D3.js became part of our application, mainly because of the capability to turn datasets into visualizations and make them interactive at the same time. The ease of usage and wide functionalities allowed the members to spend more time on the more challenging areas.

3.1.2.2 GitHub: GitHub was used to put all the files together in an organized and centralized form.

3.1.2.3 JavaScript, React and Material UI: One of the main reasons why React and Material UI were selected for this project is compatibility. The implementation with React and Material UI allowed the users to use the application on computer and mobile devices with an iOS or Android based as well. This strategy improved code reusability and allowed the team to work more efficiently.

3.1.2.4 A variety of layouts templates and design functions for JavaScript was articulated by Material UI. These layout templates allowed the members to categorize the data in a minimalistic and easily interpreted format. The team also reused a lot of the templates in the application design stage.

3.1.2.5 Jupyter Notebook: This product is part of Project Jupyter, and it is used for interactive computing across multiple programming languages.

3.1.2.6 Node.js: Node.js was implemented mainly during the data fetching phase. This library was incredibly useful, for it allowed the application to fetch the latest dataset from the provider instead of adding the dataset manually.

**3.2 Reuse of Existing Software Components**

The majority of the application was made from scratch, but it did use some templates provided by the JavaScript libraries.

**3.3 Future Plans for Software Enhancement**

Some of the plans listed on our agenda may include, but limit to the following:

* More customization options for visualizations
* Include verification/validation component to check the datasets
* A component to handle overflow errors

**3.4 User Interface Paradigms**

3.4.1 None applicable at this time.

**3.5 Hardware and/or Software Interface Paradigms**

3.5.1 Mobile device must have a touch display for user interaction

3.5.2 Personal computer must have a pointer device for user interactions and selections

**3.6 Error Detection and Recovery Options**

3.6.1 None applicable at this time.

**3.7 Memory Management Policies**

3.7.1 Memory management will be handled by the operating system, as well as the browser installed on the device.

**3.8 External Databases**

3.8.1 None applicable at this time.

**3.9 Distributed Data / Control over a Network**

3.9.1 None applicable at this time.

**3.10 Generalized Approaches to Control**

3.10.1 None applicable at this time.

**3.11 Concurrences and Synchronizations**

3.11.1 None applicable at this time.

**3.12 Communication Mechanisms**

3.12.1 None applicable at this time.

**4. System Architecture**

**5. Policies and Tactics**

**5.1 Choice of which specific products used**

5.1.1. React library: User interface

5.1.2. Material UI: User functions and design

5.1.3. Tableau Public: Data analysis

5.1.4 GitHub: File organization

5.1.5. Node.js: Fetch data automatically

5.1.6. Jupyter Notebook: Data visualization between multiple languages

5.1.7 D3.js: Create visualizations in the application

5.1.8 Tableau Desktop: Data analysis and export

**5.2 Plans for ensuring requirements traceability**

5.2.1 Most of the products are maintained by third-party organizations

**5.3 Plans for testing the software**

5.3.1 The first step was to test whether the dataset from the trusted site can be read by the line-chart function. Once successful, the window will be switched to the other visualizations to make sure the graphs are displayed correctly.

5.3.2 The second step will test the filter functions, where certain attributes from the dataset can be changed based on the user. Some of these attributes include but not limited to can be:

**5.4 Coding guidelines and conventions**

5.4.1 The application was broken down into individual components, then assigned to group members to make it functional. In the end, all components were put together.

**5.5 Coding guidelines and conventions**

5.5.1 All the related files were loaded into a .JSON file for ease of access and organizing.

**6. Detailed System Design**

**6.1 User Interface**

**6.1.1 Responsibilities**

6.1.1.1 The layout of the application is written with React, a JavaScript library specifically meant for user interfaces. This component sets the framework and defines the general layout of the application.

**6.1.2 Constraints**

6.1.2.1 Because React is a JavaScript library, the browser will need to have JavaScript enabled in order for this component to function properly. Secondly, a stable internet connection is required in order to access the library.

**6.1.3 Composition**

**6.1.4 Uses/Interactions**

The user interface is the first component users will see during their access. It will remain the same even if the user decides to use the other functions of the applications, or to view the different visualizations. Therefore, the UI will interact with all the components listed in this document.

**6.1.5 Resources**

To load the interface, the computer will utilize the CPU, memory and the React library itself. The allocation of these resources will be handled by the operating system located on the user’s device.

**6.1.6 Interface/Exports**

**6.2 Card Function**

**6.2.1 Responsibilities**

6.2.1.1 This function organizes the datasets based on their source provider. It will then display it to the users. Similar function is used for displaying the line-chart and the relevant visualizations

**6.2.2 Constraints**

6.2.2.1 The function will require the browser to have JavaScript enabled, along with a stable internet connection

**6.2.3 Composition**

**6.2.4 Uses / Interactions**

6.2.4.1 User interaction with the card function will lead to the graphical visualization function.

**6.2.5 Resources**

All resources allocation will be handled by the computer, the browser from the device, and the operating system.

**6.2.6 Interfaces / Exports**

6.2.6.1 This function will accept the dataset from the source provider, and generate a line-chart.

**6.3 Navigation Bar**

**6.3.1 Responsibilities**

**6.3.1.1 Display as part of the user interface, which act as a shortcut for user to access some of the important pages.**

**6.3.2 Constraints**

6.3.2.1 The function will require the browser to have JavaScript enabled, along with a stable internet connection

**6.3.3 Compositions**

6.3.3.1 Each navigation bar item is part of the navigation bar function.

**6.3.4 Uses / Interactions**

6.3.4.1 Interactions made here will redirect the user to the pages compiled with Card function.

**6.3.5 Resources**

6.3.5.1 All resources allocation will be handled by the computer, the browser from the device, and the operating system.

**6.3.6 Interface / Exports**

6.3.6.1 Any type of user interaction will be considered as a form of input. The output would be leading the user to the selected pages.

**6.4 Toolbar**

**6.4.1 Responsibilities**

6.4.1.1 This function will allow the users to customize some of the attributes from the dataset.

**6.4.2 Constraints**

6.4.2.1 The function will require the browser to have JavaScript enabled, along with a stable internet connection

**6.4.3 Compositions**

6.4.3.1 Each customization options will change the view of that specific visualization

**6.4.4 Uses / Interactions**

6.4.4.1 Toolbar will be an individual class, while each customization will be a subclass of the toolbar function

**6.4.5 Resources**

6.4.5.1 All resources allocation will be handled by the computer, the browser from the device, and the operating system.

**6.4.6 Interface / Exports**

**6.5 Buttons**

**6.5.1 Responsibilities**

6.5.1.1 Leads to the custom visualization function, also to confirm the user’s selection as well.

**6.5.2 Constraints**

6.5.2.1 The function will require the browser to have JavaScript enabled, along with a stable internet connection

**6.5.3 Compositions**

**6.5.4 Uses / Interactions**

6.5.4.1 This component will interact with the Graphical Visualization component, which reflects the customization based on the users.

**6.5.5 Interface / Export**

6.5.5.1 All resources allocation will be handled by the computer, the browser from the device, and the operating system.

**6.5.6 Interface / Exports**

6.5.6.1 This component will take user input and submit it to graphical visualizations component

**6.6 Graphical Visualizations**

**6.5.1 Responsibilities**

6.5.1.1 Create the visualization based on the associated dataset

**6.5.2 Constraints**

6.5.2.1 The function will require the browser to have JavaScript enabled, along with a stable internet connection

**6.5.3 Compositions**

**6.5.4 Uses / Interactions**

6.5.4.1 This function will interact with buttons and toolbar components. This component will take the user’s options from the toolbar, and confirm the selections with buttons.

**6.5.5 Resources**

6.5.5.1 All resources allocation will be handled by the computer, the browser from the device, and the operating system.

**6.5.6 Interface / Exports**

6.5.6.1 The function will take the user’s selection as input, and then change the visualization accordingly.

**6.7 Data Fetching**

 **6.7.1 Responsibilities**

6.7.1.1 This component will fetch the latest data from data provider

**6.7.2 Constraints**

6.7.2.1 The function will require the browser to have JavaScript enabled, along with a stable internet connection

**6.7.3 Compositions**

**6.7.4 Uses / Interactions**

6.7.4.1 The dataset will be interpreted by the graphical visualization component

**6.7.5 Resources**

6.7.5.1 All resources allocation will be handled by the computer, the browser from the device, and the operating system.

**6.7.6 Interface / Exports**

6.7.6.1 The component will take the dataset from content provider as input, and export it to the application

**7. Detailed Lower level Component Design**

**7.1 User Interface**

**7.1.1 Classification**

**7.1.2 Processing Narrative (PSPEC)**

**7.1.3 Interface Description**

**7.1.4 Processing Detail**

**7.1.4.1Design Class Hierarchy**

**7.1.4.2 Restrictions/Limitations**

**7.1.4.3 Performance Issues**

**7.1.4.4 Design Constraints**

**7.1.4.5 Processing Detail For Each Operation**

 **7.2 Graphical Visualizations**

**7.2.1 Classification**

**7.2.2 Processing Narrative (PSPEC)**

**7.2.3 Interface Description**

**7.2.4 Processing Detail**

**7.2.4.1Design Class Hierarchy**

**7.2.4.2 Restrictions/Limitations**

**7.2.4.3 Performance Issues**

**7.2.4.4 Design Constraints**

**7.2.4.5 Processing Detail For Each Operation**

 **7.3 Toolbar and Buttons**

**7.2.1 Classification**

**7.2.2 Processing Narrative (PSPEC)**

**7.2.3 Interface Description**

**7.2.4 Processing Detail**

**7.2.4.1Design Class Hierarchy**

**7.2.4.2 Restrictions/Limitations**

**7.2.4.3 Performance Issues**

**7.2.4.4 Design Constraints**

**7.2.4.5 Processing Detail For Each Operation**

 **7.2 Card Function**

**7.2.1 Classification**

**7.2.2 Processing Narrative (PSPEC)**

**7.2.3 Interface Description**

**7.2.4 Processing Detail**

**7.2.4.1Design Class Hierarchy**

**7.2.4.2 Restrictions/Limitations**

**7.2.4.3 Performance Issues**

**7.2.4.4 Design Constraints**

**7.2.4.5 Processing Detail For Each Operation**

**8. Database Design**

This application did not include any functions from the database.

**9. User Interface**

****

 

**10. Requirements Validation and Verification**

 The application does not have any validation or verification methods at this time.

**11. Glossary**

React: A JavaScript library that is meant for designing user interfaces.

Material UI: A React component based on Material Design by Google in 2014.

Tableau Desktop: An application meant for organizing, studying and visualizing a dataset.

GitHub: An Internet hosting service for software development and control with the use of Git.

Node.js: A JavaScript runtime environment built on Google Chrome’s V8 JavaScript engine.

Project Jupyter: A non-profit, open-sourced project born out of iPython Project in 2014. It is used to support interactive data science and scientific computing across all programming languages.

D3.js: A JavaScript library for manipulating documents based on the data.

**12. References**

12.1 Data-Driven Documents (D3) Overview

<https://d3js.org/>

12.2 Project Jupyter’s Origins and Governance

<https://jupyter.org/about>

12.3 Nodejs

<https://nodejs.org/>

12.4 GitHub

<https://github.com/>

12.5 Material UI

<https://v4.mui.com/>

12.6 Title: Software Design Document\_Draft

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

12.7 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

Appendix A: Section 6 - Continued

 User Interface – React

 Card, Navbar, Toolbar, Buttons – Material UI

 Graphical Visualizations – D3.js

 Backend Organizations – Node.js

6.1 User Interface

6.1.1 The layout of the application is written with React, a JavaScript library specifically meant for user interfaces. This component sets the framework and defines the general layout of the application.

6.1.2

Because the fact that React is a JavaScript library, the browser will need to have JavaScript enabled in order for this component to function properly. Secondly, a stable internet connection is required in order to access the library

6.1.4

The user interface is the first component users will see during their access. It will remain the same even if the user decides to use the other functions of the applications, or to view the different visualizations. Therefore, the UI will interact with all the components listed in this document.

6.1.5

To load the interface, the computer will utilize the CPU, memory and the React library itself. The allocation of these resources will be handled by the operating system located on the user’s device.

6.2 Card, Navigation bar, Tool bar and Buttons

6.2.1 The card, navigation bar, tool bar and buttons functions are implemented with Material UI, a design language/framework with React. The card function sets the layout for the organizations that provided the datasets. The navigation bar will remain the same throughout the applications, and it will serve as a shortcut for the user to access some of the contents quickly. The tool bar function will allow the user to temporarily change some of the data and see the different visualizations. In the end, the buttons function gives the layout for the user to view the different visualizations, and confirm their choices in the toolbar functions.

6.2.2 Similar to the React library, the components written with Material UI may not function properly if JavaScript is disabled.

6.2.3 The purpose of the card function is to distinguish the different datasets. Then, the application can generate the appropriate visualizations based on some of the attributes for the dataset. Next, the navigation bar will serve as a shortcut, which allows the users to quickly switch to some of most frequently used tabs. The navigation bar is a component that supports the minimalistic design stated previously above. After that, the tool bar will create a customized view of the dataset with some restrictions set by the user. Finally, the buttons function can be considered as a part of the tool bar function. The buttons will allow the user to switch to the filter, confirm the customizations, and view the dataset.

6.2.4 The functions written with Material UI will work together with the UI, because these functions are part of the UI. The UI created the view window for these functions to be in. Besides the UI, the Material UI components will work with the graphical visualizations component.

6.2.5 The components will need memory, CPU processing and internet connection to access the Material UI library. Given the application can only be accessed with a browser, the resource utilization and allocation will all be handled by the operating system and the browser.

6.3 Graphical Visualizations

6.3.1 The visualization component is responsible for generating unique visualizations based on the given attributes from each dataset. The visualizations will allow the user to have a better understanding of the situation and its significance.

6.3.2 **Specific limitations? (How many MB of RAM or an approximation of resources required.**

6.3.3

6.3.4 The component will mainly work with the toolbar, buttons and the card components in the Material UI section. Users will first access the custom view page by clicking on the “Filter” choice, made with buttons. It will then allow the user to set some boundaries and confirm the selection made with buttons.

6.3.5

6.4 Data Collection

6.4.1 The component written in Node.js is responsible for taking the up-to-date data and apply it to the visualizations.

6.4.2 If the Node.js runtime is not on the local computer, it will require a stable internet connection to access the runtime online.

6.4.3

6.4.4 The component will mainly work together with the Graphical Visualization component mentioned above. The data collection component fetches the data and then passes it to the graphical visualization for analysis and visuals.

6.4.5 The data collection component will need internet connection to access the dataset posted by the trusted sites. Then, in order for Node.js to function properly, it will need connection to use the Node.js runtime (if not downloaded onto local machine)