CHAPTER III

METHODS

This chapter will discuss the various aspects of the devised methodology that will be strictly followed. Specifically, it will discuss the research design, materials and methods, measures, application testing and simulation, system flow, and system design.

Research Design

This research project will employ an applied research design. According to Hedrick et al., (1993), unlike basic research where it is theoretical in nature, applied research design is practical and descriptively employed to allow the researcher to build knowledge and develop practical solutions for a certain research problem. In other words, the objective of applied research is to develop a product to solve a looming issue that a society is currently dealing. In this case, applied research design is a suitable research design to employ since the research project is aiming to provide a solution for the identified lack of inclusive catalog and tracking of the COVID-19 situation in the locality.

Additionally, for the development protocol, the researcher will employ a Software Development Life Cycle (SDLC) model called Agile. For the development of softwares, it is highly necessary to adopt SDLC models for it provides a compelling foundation as well as a clearly defined strategy for the development of the software applications. The most basic SDLC model adopted is Waterfall for software and web development (Chandra, 2015; Kumar Pal, 2018). However, adopting the classical waterfall model in a real-world web application development project is impractical since it is idealistic and challenging to implement (Kumar Pal, 2018). Moreover, the sequential nature of the Waterfall SDLC made it unsuitable for this project. That is why the project development methodology will adopt another SDLC model called Agile Model. The said framework is different from the expected linear sequential life cycle of the Waterfall Model.

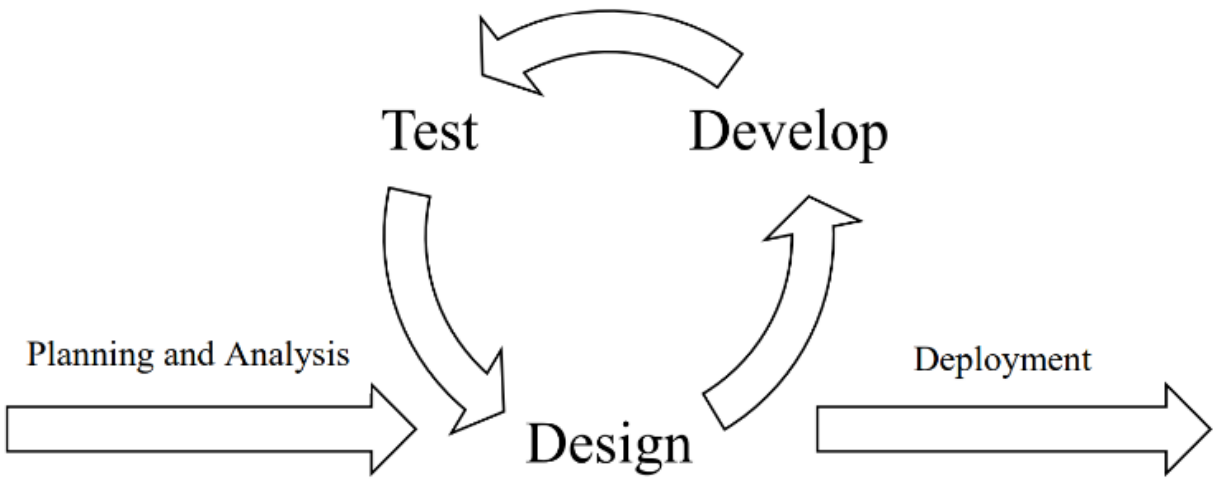


Figure 2. Agile Software Development Life Cycle Model

The primary purpose of the Agile Software Development model is to facilitate quick

project completion adaptively. The salient nature of Agile SLDC will allow the researcher to adapt to the unexpected circumstances in the development process due to its iterative and incremental nature (Figure 1). In other words, the researcher can make it up as the project goes along with the Agile Model. Whereas the Waterfall SLDC model, the researcher will structure everything before starting the project. However, with no adaptability due to its linear sequential flow, any erroneous prospects and consequences will be disregarded and not be rectified (Chandra, 2015). That is why the researcher will adopt the Agile Model since it is the most suitable SLDC model that allows the researcher to employ the advantages such as adaptability, efficiency, flexibility, incremental and continuous iteration, the high success rate with less time requirement, risk-reduction, and the elimination of cost (Dixit et al., 2020). Thus, the research project design will also adopt the Agile Model software development cycle.

Materials and Methods

Pre-development Protocol. The protocol defined for the study will be strictly adhered unto the adopted Agile Software Development Life Cycle Model. After the researcher have planned and analyzed the requirements as well as conducted a feasibility analysis, the required materials for the development, implementation, and deployment were configured and installed.

First, the researcher installed the Visual Studio Code (VS Code). It was specifically utilized by the researcher since it suffices most of the requirements and tools needed to develop the web application. Second, the latest NodeJS runtime environment was installed for the execution of the frontend and backend JavaScript code. Third, the researcher installed all the necessary frameworks through the pnpm package manager and extensions through VS Code marketplace to setup the workspace and prepare all the tools needed for the development in VS Code. Additionally, the researcher created a local directory that will contain all the project files, and initialized the version control (git init) to create a new Git repository. Lastly, the researcher connected the initialized local git repository to GitHub and pushed the existing repository to publish it online. These were the basic steps that were taken before the development and implementation of the COVID Pulse web application.

For the setup of storage and hosting of the COVID Pulse web application, the researcher installed Firebase CLI and Firebase developer tools via the pnpm package manager. Then, the researcher created a project in the Firebase Platform with free plan and enabled the hosting and storage service. After the project is ready for deployment, the researcher will build and deploy the project to the Firebase project. This procedure is to allow the COVID Pulse web application to be deployed on the internet to be publicly available to access in the browser.

Materials

Figma. It is a user interface interface and expericne prototyping program that provides a set of tools that will be used during the wireframing and designing of the COVID Pulse Web Application.

Visual Studio Code. It is an open-source Integrated Development Environment (IDE). It is a streamline IDE that has all the basic development operations needed such as version control (Git), intellisense, debugging, snippets, code completion, and refactoring, which will make the development process more efficient.

Node.js. It is a server-side JavaScript run-time environment based on the Google Chrome V8 engine. It will be utilized both on the frontend and the backend side of the web application architecture. This will be one of the foundation of the web application.

pnpm. This will serve as an installer for the necessary dependencies and packages for the project. Although NodeJS comes with built-in support for package management (npm), for this project to be time efficient and avoid any unnecessary errors during the development phase, pnpm will be used to serve as an alternative installer, and a tool to update and remove the packages that is involved during the development process.

Firebase. This is platform develop by Google that provides development services. The researcher will mainly utilize its services such as Firebase Hosting and Cloud Storage to store the necessary files and host the overall operation of the COVID Pulse web application.

Hypertext Markup Language (HTML). It is crucial for the COVID Pulse web application since it will contain the websites' basic text and hypertext contents.

Cascading Style Sheets (CSS). CSS will always coincide with HTML for the researcher to style and specify how the presentation of the User Interface will look (i.e., colors, fonts, and layout) and feel of the primary contents of the COVID Pulse web application.

JavaScript. This programming language is a dynamic client-side scripting that will allow the researcher to make the COVID Pulse web application include more functionality, responsiveness, and dynamic features.

Tools, Libraries, and Frameworks. The dependencies and devDependencies that will be implemented in the COVID Pulse project can be found in the package.json file found in the repository. It contains the metadata relevant to the COVID Pulse project repository. It will be used for managing the COVID Pulse project's dependencies, devDependencies, scripts, and version.

VueJS. It is a progressive JavaScript framework used to build web interfaces and one-page applications. In other words, it will allow the researcher to save time during the development since it is a progressive JavaScript framework that allows the process to be smooth and easy with a shallow learning curve. Furthermore, it is chosen since it is a suitable lightweight, flexible,

modular, and highly performant framework.

ChartJS. It is an open-source JavaScript data visualization library that will be adopted. Unlike the other leading data visualization library such as D3.js, Chart.js is straightforward, requires less effort, and sufficed the bare minimum requirement of generating data graphics to develop the COVID-19 dashboard. Furthermore, it will also be paired with Google Data Studio as a complementary for converting the COVID-19 data into reports.

AxiosJS. It is an promised based HTTP client that will be used in the NodeJS. This will mainly be used for the retrieving API data from services through GET requests methods.

Data API Sources

The data dashboard will acquire the COVID-19 aggregated epidemiological data from various APIs in terms of Local, Regional, National, and Worldwide. Specifically, the data source will be from the following repositories:

JHU CSSE COVID-19 Data. It is a COVID-19 data repository collected, provided, and operated by the Center for Systems Science and Engineering from Johns Hopkins University. It is publicly available for everyone to be accessed from the GitHub JHU CSSE repository.

WHO COVID-19 Data. It is the official COVID-19 data source aggregated by the World Health Organization and is distributed by comma-separated values (CSV) files.

DOH COVID-19 Data. The official COVID-19 data source aggregated by the Department of Health can be accessed through the DOH Data Drop.

Designing the Web Application

The designing stage is crucial for the development of the COVID Pulse. In this stage, the researcher will identify and describe the web application's features, operation, and specification to establish the intended objectives. The system designing of the COVID pulse will consist of various design considerations and concepts. Additionally, it conceptualizes and offers good visual and descriptive prospects about the web application and its system aspects to allow the final version to be consistent with design structures as described initially in the proposed system architecture models. Hence, this stage is necessary since it will allow the researcher to implement and code the devised and analyzed prospects in the Planning and Analysis phase through a programming language. Lastly, the web application system design of COVID Pulse will be divided into three segments: Frontend, Backend, and APIs.

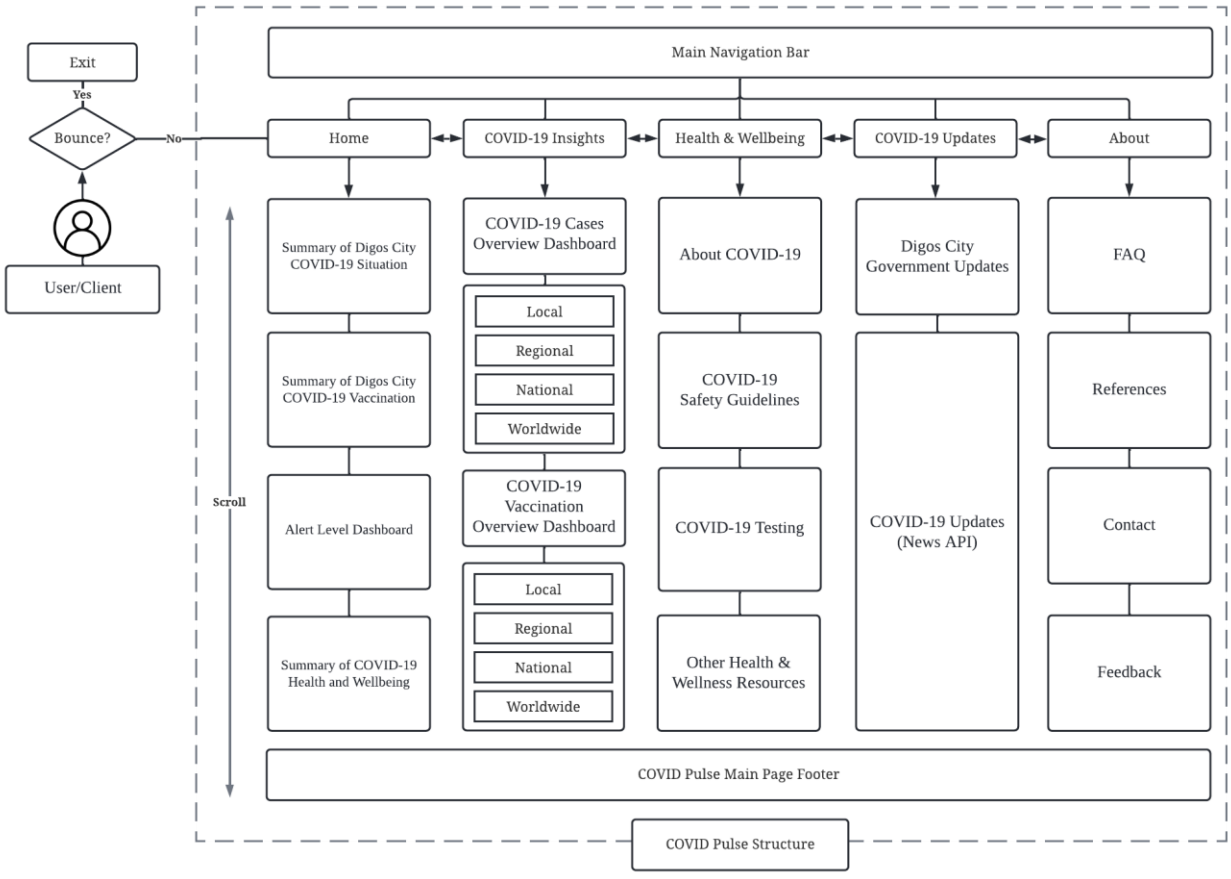


Figure 3. *COVID Pulse Web Application Sitemap Structure*

Home - The home webpage will act as the website's beginning point. The default page loads when the target users, such as the Digoseños, visit the COVID Pulse website.

COVID-19 Insights - This webpage will contain the main objective of this project. The elaborated COVID-19 dashboard that visualizes the COVID-19 Cases and Vaccination per segment will be embedded in the said webpage.

Health & Wellbeing - This webpage will contain information about the SARS-CoV-2 virus, COVID-19 safety guidelines, COVID-19 testing, and other health-related resources.

COVID-19 Updates - This webpage will contain the essential updates from the Digos City government, such as the Alert Level ordinance, and will also contain the COVID-19 related news articles for the Digoseños to be constantly updated and informed.

About - This web page is to inform the web application visitors about the COVID Pulse's details and the web application's critical operations.

Frontend Prototyping

The initial but most crucial phase of the design stage of the development process is prototyping the COVID Pulse web application. The researcher will be able to ideate the reference for the end product, which allows the development process to be convenient and reduce the cost and time as it provides a comprehensive high-level reference and overview of the final output. Furthermore, the prototyping phase will allow the researcher to make quick necessary modifications and be flexible with the User Interface and User Experience design. Through wire framing, the prototyping will be done through Low-Fidelity and Hi-Fidelity prototypes (Figures 4

and 5). Although sketching is often part of the prototyping procedure, it was not included since it is deemed unnecessary.

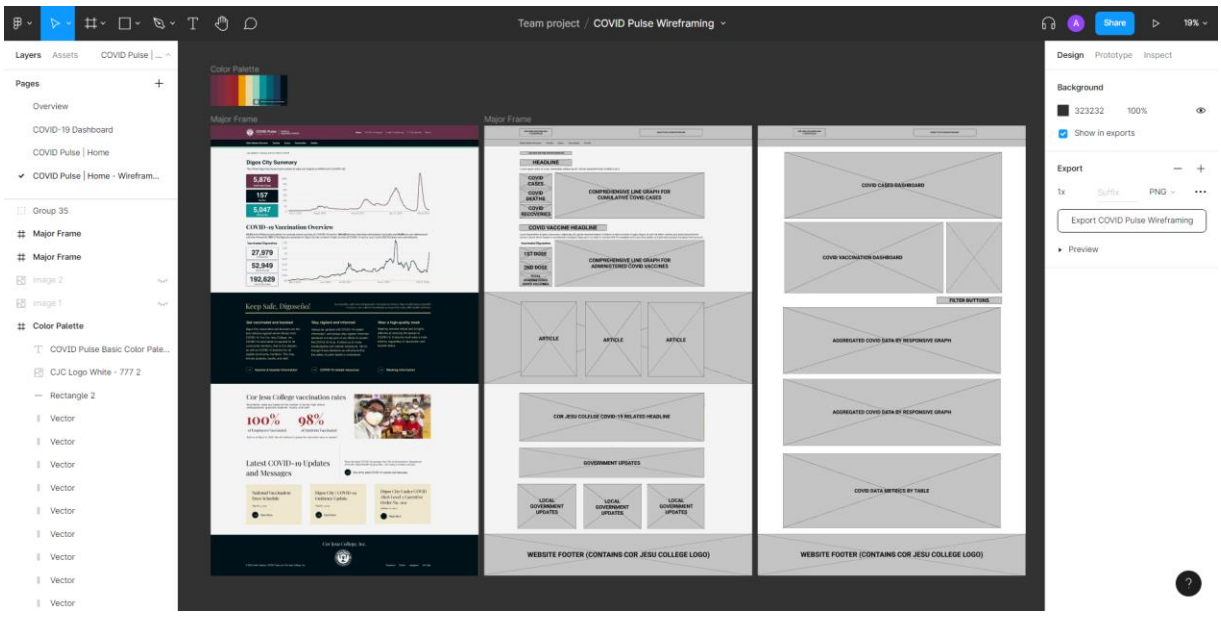


Figure 4. *Designing the UX/UI of COVID Pulse Web Application using Figma*

Low-fidelity (Lo-Fi) prototyping is essential for the researcher to quickly conceptualize the COVID pulse's design features. However, the appearance design will be disregarded in low-fidelity prototyping (Figure 4). This prototyping technique will allow the researcher to convert the high-level design concept of the web application into a testable functional prototype and know each purpose of the components. Then, the low-fidelity prototype will be anchored to the next prototyping stage, which is the high-fidelity prototyping technique. See Figure 6 for a hi-fidelity sample of COVID Pulse.

The outer layer of the web application project that the Digoseños (Users) see and interact with is the Frontend, also known as the client-side. Specifically, it is the visual elements such as the User Interface (UI) and User Experience (UX) designing of COVID Pulse. The backend layer is scoped on the server-side of the web application, in which the primary purpose is to make sure everything of the web application is functional. Also, it is the part where the clients of the COVID Pulse will not interact and cannot be interacted with users. The frontend layer will consist of the languages that are the fundamental pillars for Web Development: HTML, CSS, and JavaScript. The researcher will adopt JavaScript and other frameworks and libraries for the backend. Additionally, the researcher will implement tools, frameworks, and libraries such as Tailwind CSS, Vue.js, Chart.js, Cypress, Lighthouse and Firebase during the COVID Pulse web application development.

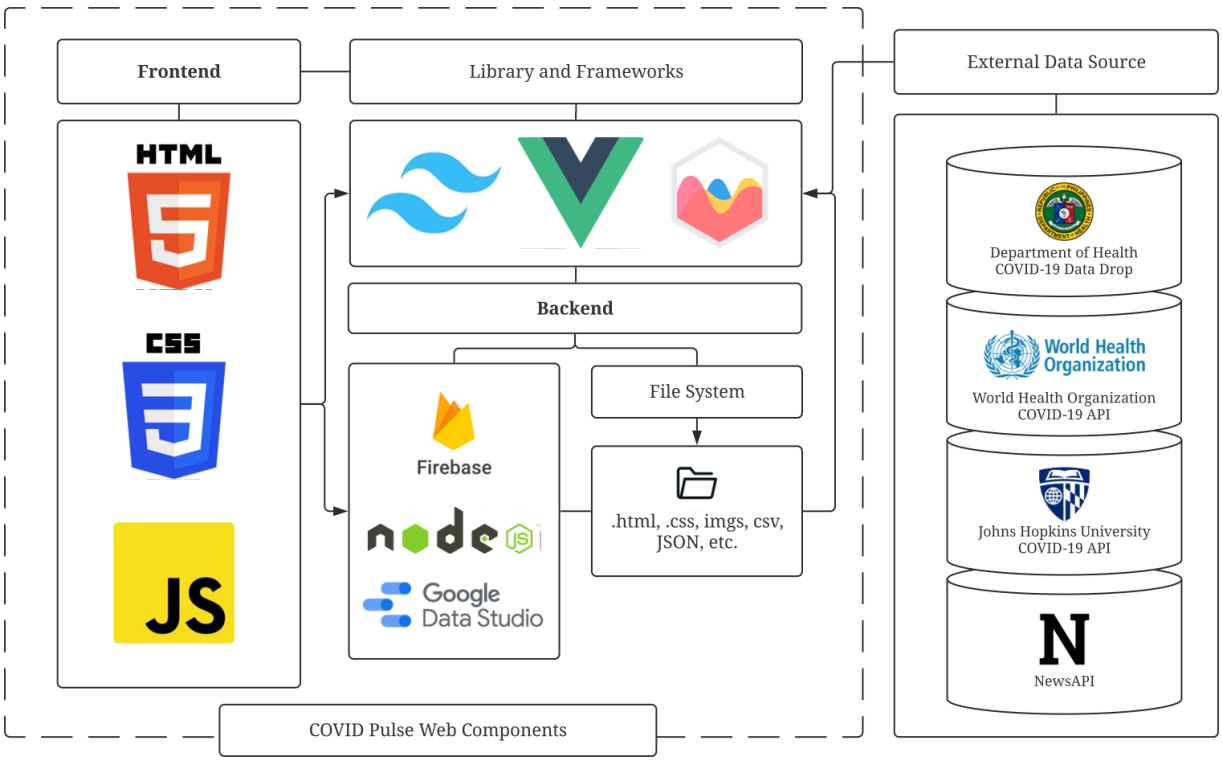


Figure 7. *High Level Architecture, Frontend, Backend, and APIs components*

Web Application Design

The technological architecture of a system is crucial since it determines its functionality and operational stability. In this study, the COVID-19 Pulse web-based platform will be designed and developed to provide relevant COVID-19 related information catalogue and tracking in realtime. Specifically, the proposed platform will provide dynamic and real-time comprehensible epidemiological indicators, a user-friendly interactive user experience and interface, brief narratives to summarize and interpret displayed data, and relevant news updates related to COVID-19 situation. Since the COVID Pulse web application will be developed using Vue, its structure will be a Single Page Architecture (SPA). The web application architecture is shown in Figure.

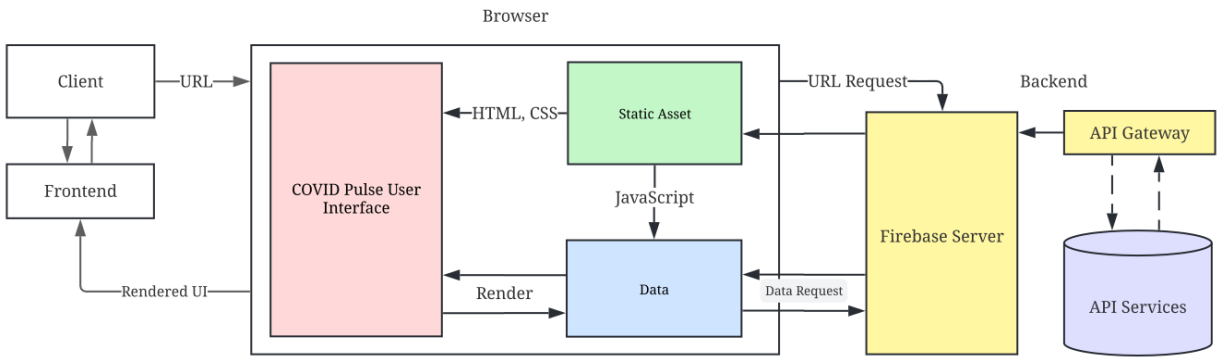


Figure 8. *Single Web Page Architecture (SPA) of COVID Pulse Web Application*

SPA will be adopted because it allows the web application to update its contents dynamically. The web pages in the web application will then be divided and mapped to the pages in each routes. This method is called routing, which will be integrated in the VueJS through Vue router.

Singe Page Application (SPA) architechture will be achieved through the implementation of Vue Router. It is a feature that enables the users to navigate between pages without refreshing

the entire web application. This makes the navigation experience easier. Instead of requesting resources from the server in every request or navigation, SPA application architechture will allow the users to continuously interact with the COVID Pulse web application by dynamically rewriting the main page without reloading the web pplication itself.

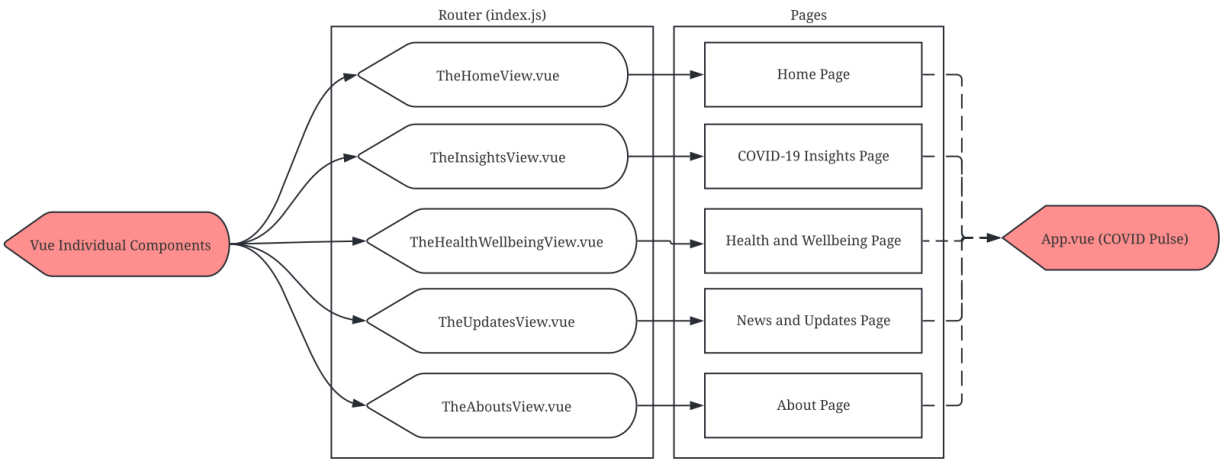


Figure 9. Flow Chart of the Routing Structure for Each Vue Page

Every individual component in a single view page component will be merged, as seen Figure. Then, in each view component, it will be linked to every page that is associated with the given address. In this case, the view pages, which contains all the individual components, is linked to the pages via linking and routing them. Router (router-view) is the view that will render the components that are merged.

Functional Design

There are two major functionality that are proposed in the COVID Pulse web application: Catalogue of COVID-19 Related News and Tracking of the COVID-19 Epidemiological Insights. The former will be achieved by implementing a third-party data APIs request using Axios.js that scrapes articles from the internet. In the event of the user visiting the COVID-19 update page, an HTTPS API will execute await/async function that has a pre-determined instructions such as method, URL, parameters, and headers. If the API Service, in this case NewsAPI, will respond with an OK status response, then the retrieved data from the response will be parsed using Day.js for the date and time and binded into the Vue News section component. Additionally, the way the data is binded will be the v-for list rendering directive. The v-for directive will assign a unique syntax of item which will be used as an index in each element iterated in the array retrieved in the NewsAPI response. An example of this process is shown in Figure and as well as the snippet code is provided in Appendix I.

The flowchart shown in Figure 11 shows the process of how the COVID-19 related news data is streamed into the News Section component in the event of the user loads the COVID-19 updates page. Clicking the COVID-19 news page in the navigation will triger an HTTP GET Request that will load the COVID-19 Updates component from the Firebase while executing the

API request simultaneously. The API Request contains the Async/Await Function that will set the promise preconfigured in the function. If the API Response returns an Error, then the console log will return and no data will be feeded into the News Section component. However, if the API responded with an OK status (200), then the response will contain a news data, in json, that fulfi;ls the promise. Then, the API request function will then feed the retrieved News data to the COVID-19 News Section component. Lastly, the COVID-19 News Section will be rendered to the user client which display the COVID-19 related articles. A specific example of this process is also shown in Figure 10 where the headers and parameters are explicitly defined.

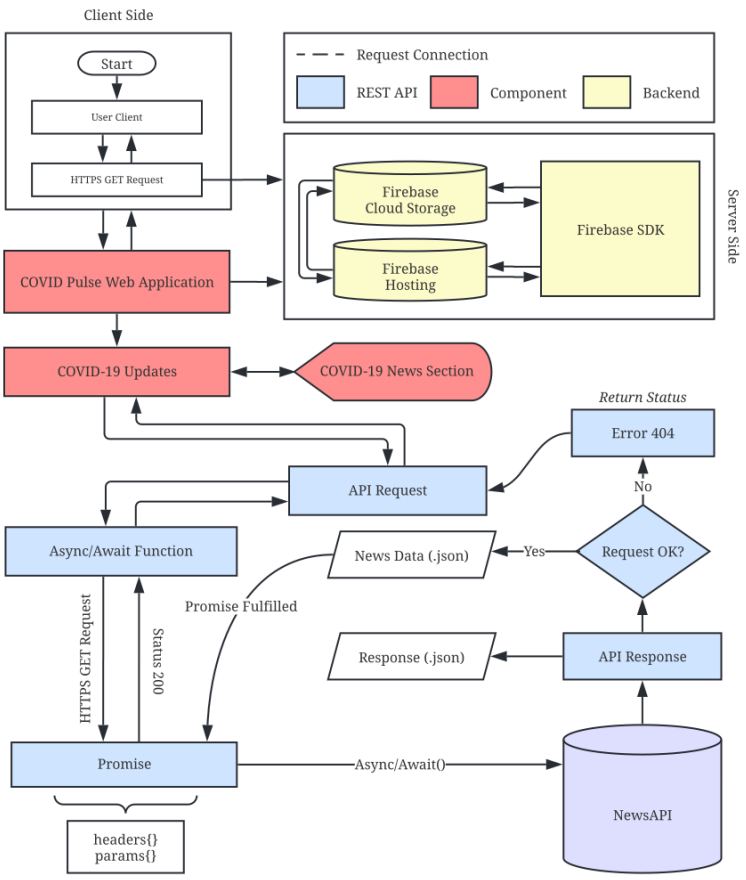


Figure 11. *Flowchart of the retrieval of news data from NewsAPI to COVID-19 Pulse Updates Page Component*

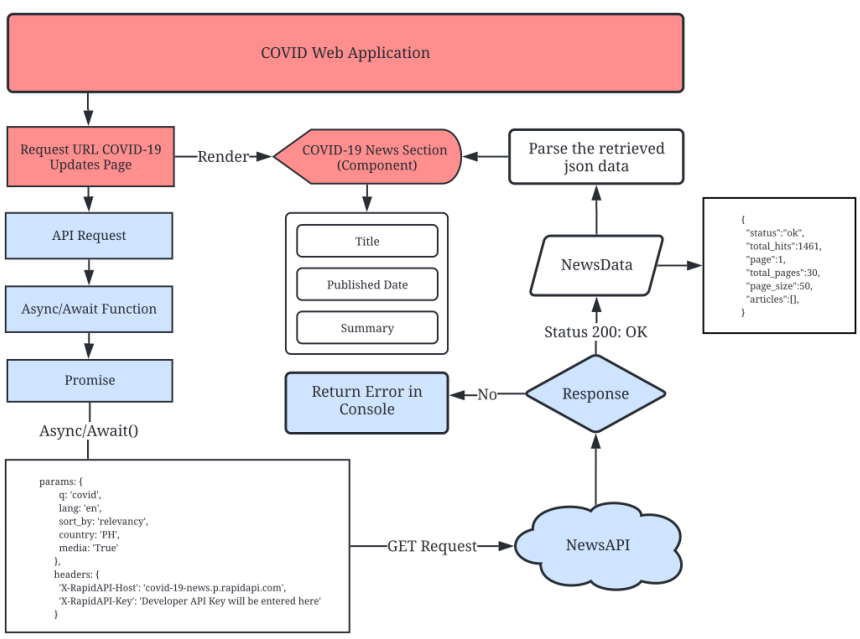


Figure 10. Example API Request to NewsAPI then parsing and binding the response data to the COVID-19 News Section Component

The tracking of the COVID-19 Epidemiological Insights in the web application will still implement a COVID-19 data source repository through Axios and the same binding technique will also be used to pass data into the components of the COVID-19 dashboard page. In addition, another COVID-19 dashboards will be embedded to the COVID-19 Insights Page components.

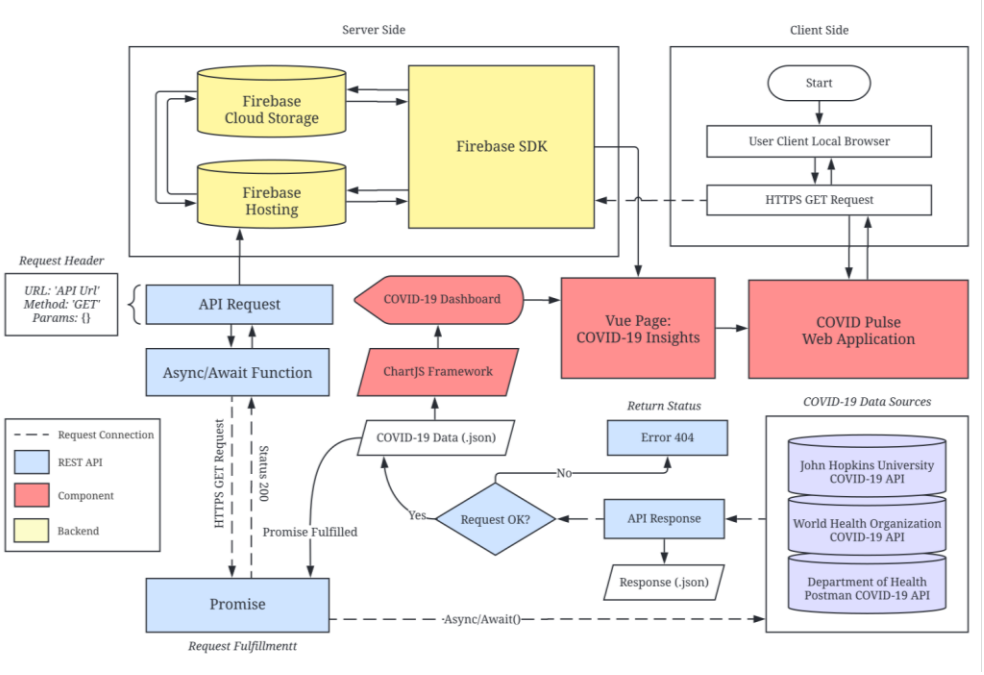


Figure 12. *Flowchart of the COVID Pulse Dashboard retrieval of data from COVID-19 data sources*

Figure 12 shows the process of rendering the COVID-19 Dashboard. First, the user will trigger the HTTPS GET request that will trigger the API Request that contains the Await/Function. Next, the promise is set based on the defined parameter in the request header. The API services in the COVID-19 Data Sources will respond. If the request will return an error, then no data will be streamed in to the ChartJS framework. However, if the request OK (200) then the COVID-19 data in the form of JSON will be feeded into the ChartJS framework, which will be then rendere into th COVID-19 Dashboard component. Lastly, after the API request procedure, the rendered dashboard will then be returned in the web application.

Development Stage

This stage involves the development of the COVID pulse through the implementation and coding of the designed project. In other words, this is the primary stage in the realization of the COVID Pulse web application design and translating it into a source code. Each module that will be designed in the designing stage by the researcher will be implemented and coded. After the development stage, the researcher will test the module functionality and determine whether it is appropriately working through end-to-end (E2E) testing. Each development phase will focus on the three segments: Frontend, API (Middleware), and Backend.

Frontend and Backend Layer. The outer layer of the web application project that the Digoseños (Users) see and interact with is the Frontend, also known as the client-side. Specifically,

it is the visual elements such as the User Interface (UI) and User Experience (UX) designing of COVID Pulse. The backend layer is scoped on the server-side of the web application, in which the primary purpose is to make sure everything of the web application is functional. Also, it is the part where the clients of the COVID Pulse will not interact and cannot be interacted with users.

The frontend layer will consist of the languages that are the fundamental pillars for Web Development: HTML, CSS, and JavaScript. The researcher will adopt JavaScript and other frameworks and libraries for the backend. Frameworks and libraries such as Tailwind CSS, Vue.js, and Chart.js will be implemented in the COVID Pulse web application.

Web Application Testing and Simulation

The testing stage is part of the development process. The project will adopt a type of functional test called End-to-End (E2E) testing, specifically an automated Horizontal E2E. This type of testing involves testing the entire software, or in this case, website web application, from start to end and will coincide with the user flow [Hamilton, 2019. Its primary goal is to test the developed project to validate if everything from all the integrated units is behaving as expected. E2E testing is mainly done from the aspect of the end-user by simulating actual real-world user experience and verifying the entire system.

The vertical E2E test breaks down the COVID Pulse structure into various segments. Each segment of the web application will be individually tested and analyzed in a hierarchical order. Unlike the vertical testing, the testing phase of the project COVID Pulse web application does not involve breaking down the codes into individual units before testing it. There are many benefits of E2E testing, such as wide testing coverage, ensuring system consistency, reducing time and cost, and detecting errors and bugs during the development phase (Rajkumar, 2022). E2E testing will also allow the researcher to verify the web application flow, which will alleviate the project's potential risks. Therefore, the researcher will adopt the Horizontal E2E testing during the iteration and incrementation of the development process (Figure 9) since it is an efficient, suitable, and reliable testing methodology.

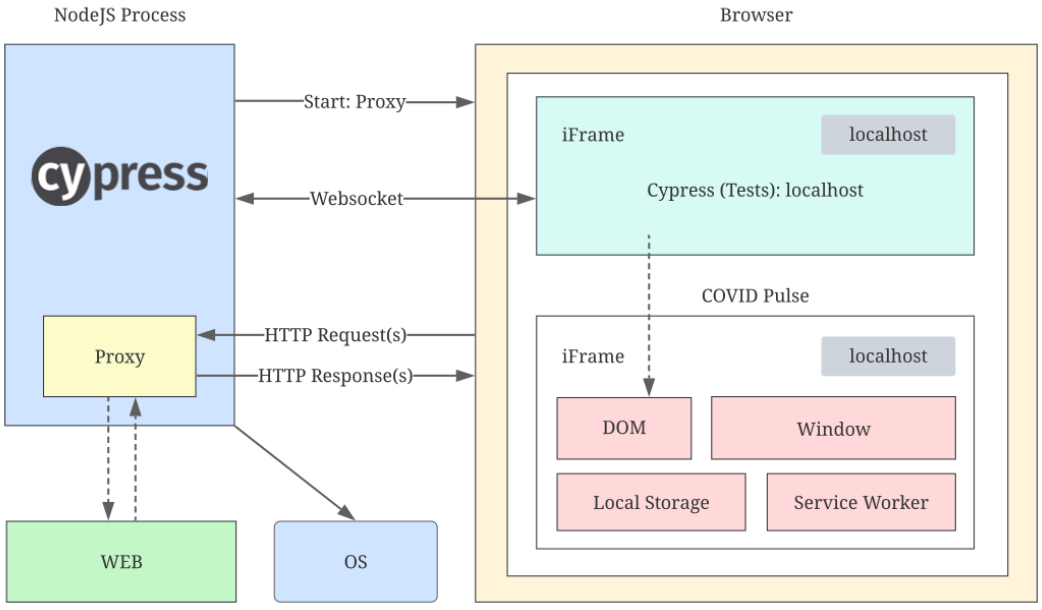


Figure 13. *Cypress Testing Architecture for COVID Pulse*

High Level Architechture Diagram of the Adopted Cypress Testing

The researcher will adopt a testing solution called Cypress, which is a JavaScript Framework. First, the Cypress is opened through the VS Code terminal, which will open the Cypress Test Runner and creates a folder in the root of the project files that will contain all the essential testing codes. Next, the test setup is configured by creating a test suite in the integration folder. In the test suite, the researcher described the test commands and assertions programmatically. Every essential component of the web application will be tested to ensure the existence, functionality, and flow of behavior functions as expected.

Web Application Interface and Components

The interface of COVID web application has multiple components. The figures provided displays the component user interfaces in every pages. As shown in Figure 14, it shows the summary dashboard that provides all the essential and relevant COVID-19 epidemiological insights for the present date. However, note that some of the displayed numerical data are at risk of being outdated due to either the COVID-19 API data service have not updated its database or the specific facilities still have not reported their COVID-19 cases to the Department of Health.

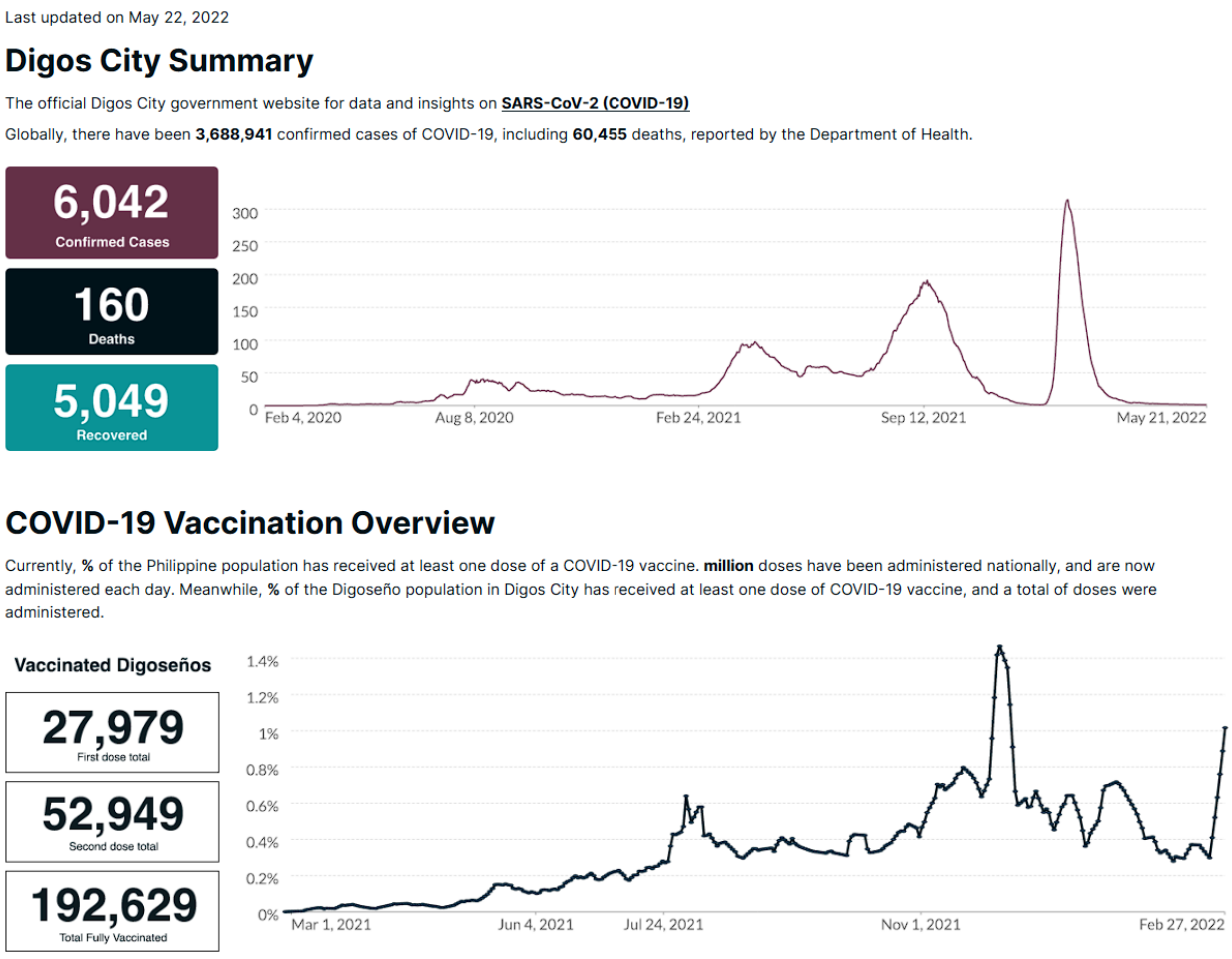


Figure 14. *COVID-19 Dashboard Summary in the Home Page*

Figure 15 shows the preliminary user interface design in the COVID Pulse home page. The users will be given a summary card of the essential contents contained in the COVID-19 Health and Wellbeing and COVID-19 Updates and Messages summary section

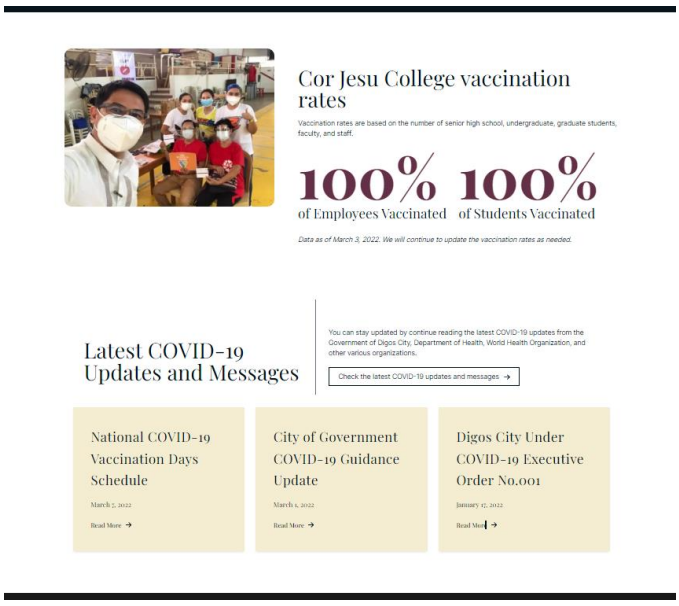


Figure 15. *COVID-19 Updates and Messages summary section in the Home Page*

In Figure 16, it is the user interface design of the COVID-19 Health and Wellbeing. It is where the users can access the essential COVID-19 guides related to the maintaining health behaviours during the COVID-19 pandemic.

Figure 16. *COVID-19 Health and Wellbeing summary card in the Home Page*

Figure 17 shows the salient dashboard feature of the COVID Pulse. In the figure shows the COVID-19 dashboard that sourced its data from the JHU CSSE repository. It can provide all the necessary epidemiological data that is needed for the users to get updated.

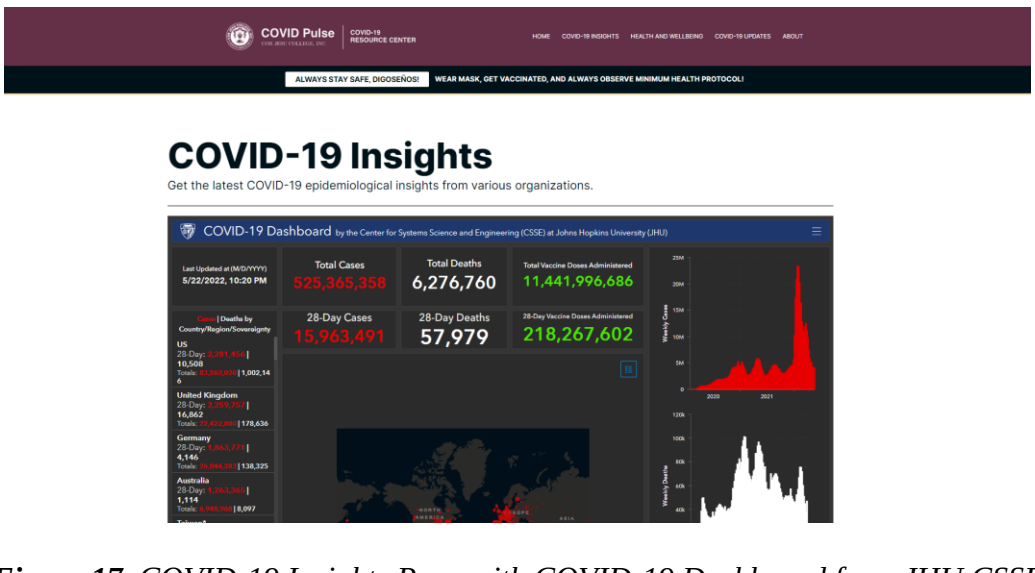


Figure 17. *COVID-19 Insights Page with COVID-19 Dashboard from JHU CSSE*

Figure 18 shows the COVID-19 Dashboard component which sourced its data from the World Health Organization. It has an interactive cataloguing feature that divides the epidemiological data per category.

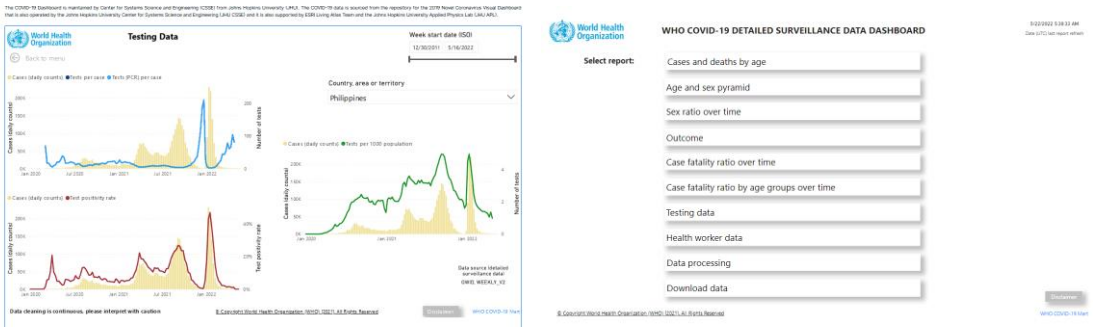


Figure 18. *COVID-19 Insights Page with COVID-19 Dashboard from WHO*

Figure 19 shows the DOH Dashboard that provides the status of the available facilities in the Philippines. It also has an interactive control that users can use for specifying data that will be displayed in the dashboard per Region, Province, and City/Municipality.

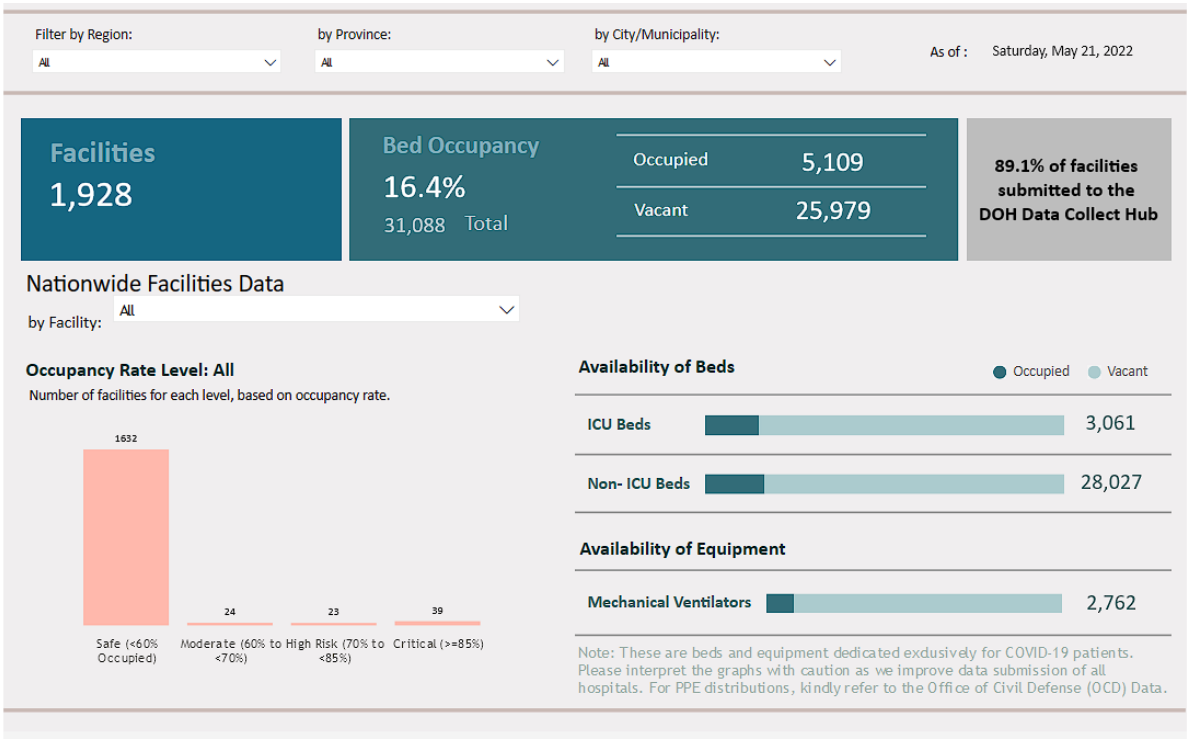
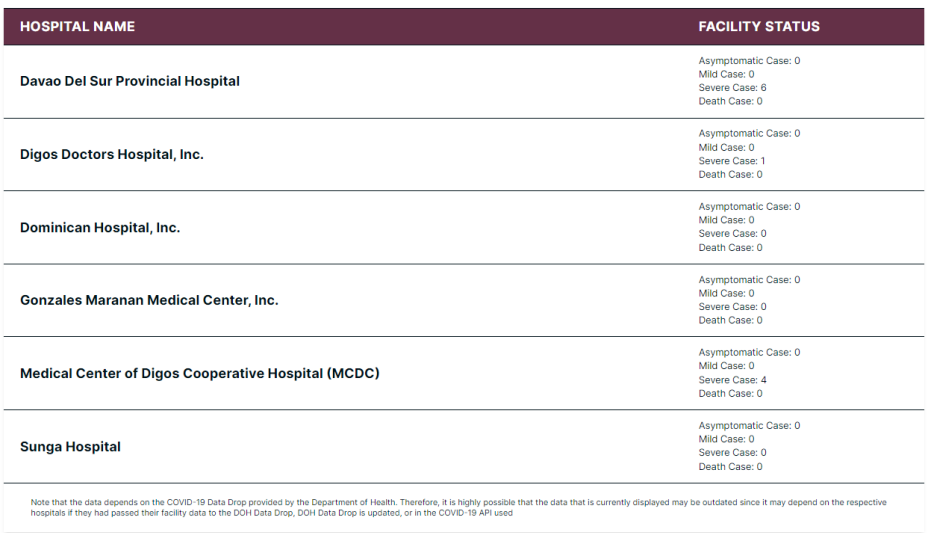


Figure 19. Department of Health COVID-19 Facility Status Dashboard

Figure 20 shows the specific number of COVID-19 cases in the COVID-19 Facilities that is located in Digos City. In every facility in Digos, most report their data according to the COVID-19 cases that is asymptomatic, mild, severe, and death.



HOSPITAL NAME	FACILITY STATUS
Davao Del Sur Provincial Hospital	Asymptomatic Case: 0 Mild Case: 0 Severe Case: 6 Death Case: 0
Digos Doctors Hospital, Inc.	Asymptomatic Case: 0 Mild Case: 0 Severe Case: 1 Death Case: 0
Dominican Hospital, Inc.	Asymptomatic Case: 0 Mild Case: 0 Severe Case: 0 Death Case: 0
Gonzales Maranan Medical Center, Inc.	Asymptomatic Case: 0 Mild Case: 0 Severe Case: 0 Death Case: 0
Medical Center of Digos Cooperative Hospital (MCDC)	Asymptomatic Case: 0 Mild Case: 0 Severe Case: 4 Death Case: 0
Sunga Hospital	Asymptomatic Case: 0 Mild Case: 0 Severe Case: 0 Death Case: 0

Note that the data depends on the COVID-19 Data Drop provided by the Department of Health. Therefore, it is highly possible that the data that is currently displayed may be outdated since it may depend on the respective hospitals if they had passed their facility data to the DOH Data Drop, DOH Data Drop is updated, or in the COVID-19 API used

Figure 20. *COVID-19 Insights Page with Facility Status with data source from DOH*

Figure 21. *COVID-19 Updates Section with News Section that contains COVID-19 related articles*

Figure 21 illustrates the preliminary COVID-19 News section of COVID Pulse. Every article card provides details retrieved and parsed from NewsAPI such as the news title, news excerpt, date of published, and the provided article media.