Auxillia Diagnostics

Medical Diagnostic and Reference Tool

Author: John Carlo M. Ramos

Supervisor: Susan Reardon
Second Reader: Marian McDonnell

Dissertation submitted in partial fulfilment for the degree of BSc (Hons) in Computing Multimedia Systems/Web Engineering

07/05/2021

Declaration of Authorship

The incorporation of material without formal and proper acknowledgement (even with no deliberate intent to cheat) can constitute plagiarism.

If you have received significant help with a solution from one or more colleagues, you should document this in your submitted work and if you have any doubt as to what level of discussion/collaboration is acceptable, you should consult your lecturer or the Course Director.

WARNING: Take care when discarding program listings lest they be copied by someone else, which may well bring you under suspicion. Do not to leave copies of your own files on a hard disk where they can be accessed by other. Be aware that removable media, used to transfer work, may also be removed and/or copied by others if left unattended.

Plagiarism is considered to be an act of fraudulence and an offence against Institute discipline.

Alleged plagiarism will be investigated and dealt with appropriately by the Institute. Please refer to the Institute Handbook for further details of penalties.

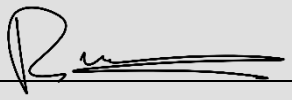
The following is an extract from the B.Sc. in Creative Computing (Hons) course handbook. Please read carefully and sign the declaration below

Collusion may be defined as more than one person working on an individual assessment. This would include jointly developed solutions as well as one individual giving a solution to another who then makes some changes and hands it up as their own work.

DECLARATION:

I am aware of the Institute's policy on plagiarism and certify that this thesis is my own work.

Student: John Carlo M. Ramos

Signed 

Failure to complete and submit this form may lead to an investigation into your work.

Abstract

Medical staff and students find that researching information about symptoms may take longer than expected. Within the medical and educational industry, this may be an issue due to the fast-paced nature of their field. The aim was to design and develop a reliable system that provides medical personnel the fastest way to get the information that they require. The steps involved in the development of this project were researching the requirements, designing the system, implementation of features and testing the application via functional and user testing.

Acknowledgement

I would like to express my upmost gratitude to my supervisor Susan Reardon for the guidance and assistance during the design and development process. Her experience in design had been an inspiration to help me improve my current skills while also allowing me to learn new ones. She had taught me many valuable things that I could bring with me throughout my career.

I would also like to thank my dear friends of mine in class 2017 who I consider all as my brothers. For four years, we went through thick and thin, especially during the pandemic crisis. When I had struggled and lost my way, they were all there to pick me up and set me straight again. For that, I cannot thank them all enough, especially to Eoan O'Dea. If it was not for his patience and professionalism, I will not be where I am today.

Last but not least, I would like to mention my late mother-in-law, Geraldine Casimpoy. She was the reason why I chose to do a medical tool for my final year project. Not only was she an inspiration to my sister, her sons and her granddaughter, but she was also known throughout the whole South Dublin's medical and Filipino community. Thank you for everything, *tita*.

Table of Contents

1 Introduction

The aim of this project was to provide both medical staff and students with a system that will ensure optimum workflow during their symptom research. This application has been designed and developed to have a quick search functionality for the symptom information, note-taking and bookmarking capabilities as well as providing easy communication between multiple users. In addition, the user will have the ability to create and publicise their own knowledge should they discover a development. Considering that this project is dedicated for medical staff and students, the application area is targeted for the medical and education industry.

This documentation outlines the requirements to develop the system, the research findings for interaction and user-centred design, the iteration of the design as well as the structure of the project, the implementation of functionalities and features, functional and user testing.

2 Requirements and Feasibility

2.1 Introduction

The aim of the project was to design and develop a medical diagnostic application. It serves as a reference tool for students, doctors/nurses, general practitioners and/or emergency workers. It would allow medical staffs to look up symptoms, determine possible causes and recommend appropriate diagnosis. The objective for this project is to design an application to efficiently and reliably help the medical staffs to enhance their workflow as well as to potentially achieve medical and surgical intervention. Alternatively, should the user be a medical student, the objective is to grant them accurate medical data.

Data about the symptoms, causes and diagnosis are generated via API. The option for users to input their own knowledge may be considered as well. This will allow users to publicly post comments on the information page, which would lead to a potential user account feature. Users will have the option to bookmark their information as well as permission to customise on how they would like to organise it. This application will be designed to have a contemporary, clean and simple architecture to provide users the ease of use. It will also be developed for all devices (i.e. iOS, Android and web).

The framework that was used is as follows: MongoDB Atlas, Express.js, Node.js and a symptom checker API for the backend, and React.js and Atlaskit – Atlassian component library for the frontend. Throughout the progress of this project however, original component library was replaced with Google Material UI. This is discussed in more detail in the *Design Chapter*.

2.2 Requirements Analysis

2.2.1 Existing Applications

Included within the *Requirements and Feasibility* chapter are research done on a few existing applications. The findings of this research are used to determine the functionalities and features of those applications which were then used to amplify the strengths and reduce the weaknesses of the medical diagnostic tool project.

Symptomia (iOS exclusive):

This iOS exclusive application is a medical diagnostic tool that is designed to meet the needs of medical professionals and students. It is an application that allows for quick access to medical information based on the most common symptoms. Features included are symptom data, fast search function, and bookmarking feature (Figure 1 and 2). It is also worth noting that information within this application is focused on the patients.

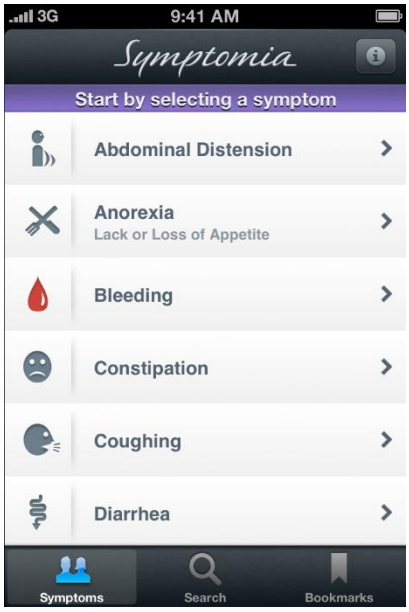


Figure 1 - Dashboard

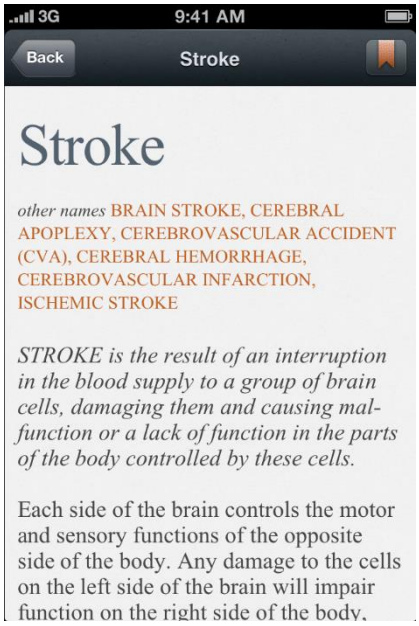


Figure 2 – Information Page

Advantages:	Disadvantages:
• Users are able to bookmark information. • User-friendly interface. • A search bar to allow the user to find information quickly.	• Only available on iOS. • Limited information and some may not be updated. • Users cannot input their own information. • No illustrations to grant users further understanding.

Critical Care ACLS Guide (Android exclusive):

This Android exclusive application is used to give users immediate access to critical information in a relatively simple to use app with rich content and vivid diagrams. Features included are data containing Advanced Cardiovascular Life Support (ACLS) and a medical calculator to allow the user to check ACLS drug doses, interpret EKGs and search patient medications (Figure 3 and 4).

Figure 3 - Dashboard

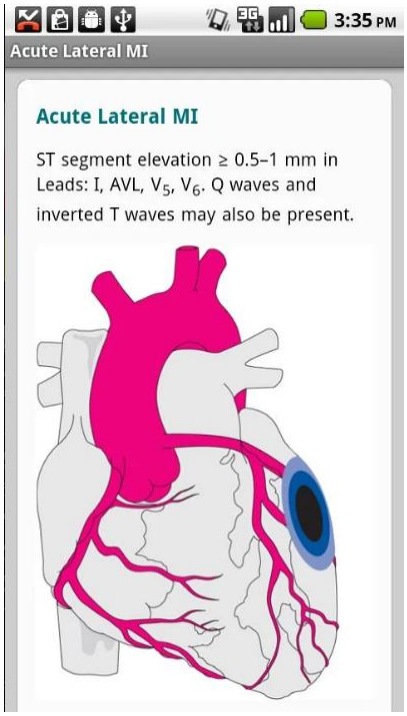


Figure 4 – Information Page

Advantages:	Disadvantages:
• Users are able to bookmark information. • Expanded search capability with an intensive amount of detailed information based on ACLS. • Information includes vivid diagrams to give the users further understanding.	• Only available on Android. • Information specific to only ACLS. • Reviews have stated that the available information have not been updated. • UI is outdated in comparison to contemporary designed apps. • Users cannot input their own information.

Eponyms (iOS exclusive):

The term *Eponym* is used to describe a symptom that is named after someone (e.g. Shiel syndrome). This application provides short descriptions for an approximate 1,800 common and obscure medical eponyms. Features included are 26 categories for eponyms, a "learn mode" which allows the user to treat the information equivalent to flash cards for studying, a responsive search bar and a dedicated database from Andrew J. Yee (http://www.eponyms.net/). See Figure 5 and 6 for an example of the application.

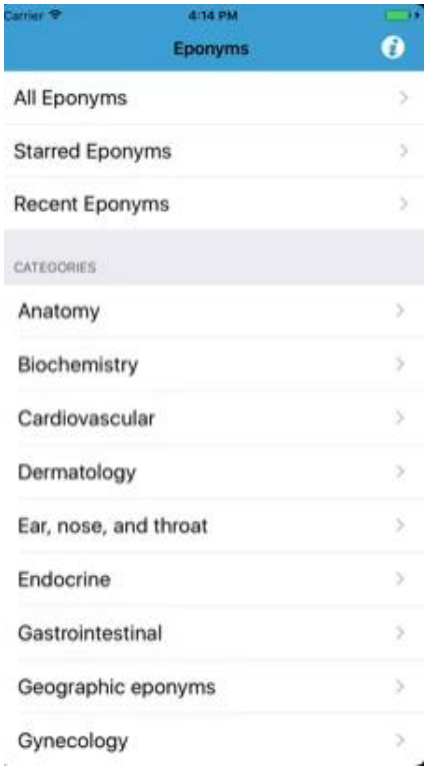


Figure 5 - Dashboard

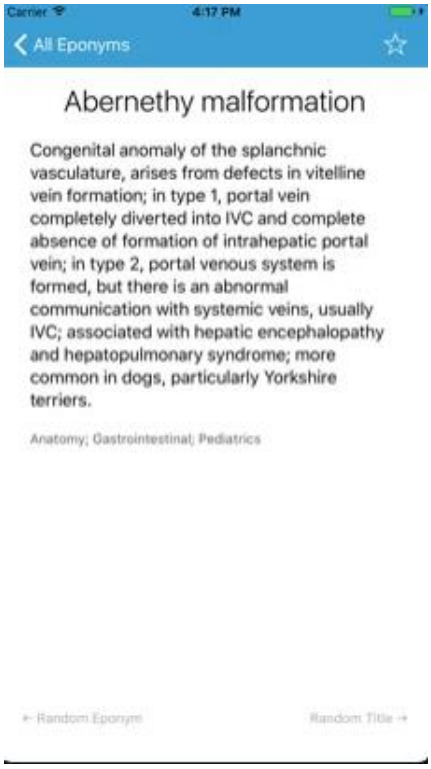


Figure 6 – Information Page

Advantages:	Disadvantages:
• Uses a dedicated database to get information. • Includes a simple yet intuitive feature for efficient learning. • A lightweight app. • Includes a search history. • Can be used by students and professional healthcare workers.	• Only available on iOS devices. • While the UI is simple for lightweight purposes, the UI is too basic. • Users cannot input their own information.

Pedi STAT (iOS exclusive):

This application is considered to be a rapid reference tool for medical staff caring for paediatric patients in the emergency or critical care environment. Features included are information and calculator instruments that are specific for the age and weight of the paediatric patients, references of age specific normal vital signs and colour-coded organisation for ease of access. See Figure 7 and 8 for an example of the application.

Figure 7 - Dashboard

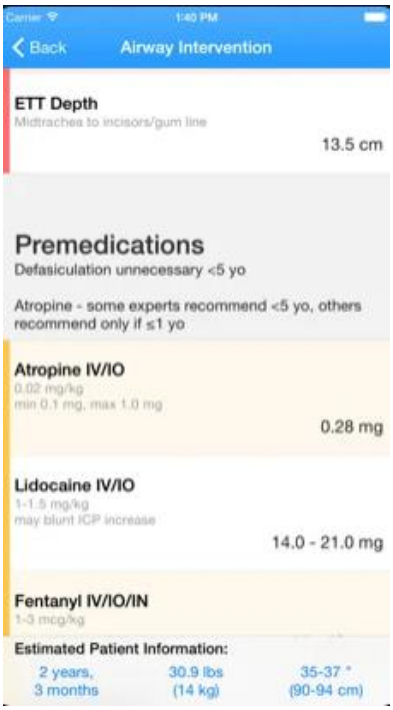


Figure 8 – Information Page

Advantages:	Disadvantages:
• Clean, modern and organised design to create satisfying UX. • A versatile reference tool with a simplistic medical calculator. • Can be used by students and medical staffs.	• Only available on iOS devices. • No customisation features (i.e. cannot save and organise information the way that the user would desire). • Users cannot input their own information.

2.2.2 User Profiles

User profiles are built to help the designer understand the requirements of the user. For this project, interviews were conducted with potential users to determine their requirements. Personas (i.e. fictional characters) were also made to understand the users' needs as well as identify who the relevant users are (*What is Persona Development and Why is it Important?*, 2016). In conjunction with a persona, a scenario has also been made. These are fictional stories that the persona acts out and are used to predict on how certain types of users will interact with the application in a given situation.

Interviews:

Due to the focus of this project in combination with the COVID-19 pandemic, interviewing multiple medical students and staff were not achievable in the time of writing this documentation. However, an interview with one Intensive Care Unit (ICU) registered nurse and a former medical chemistry student were documented as seen in *Appendix E - Interviews*. These participants had decided to not allow their names to be disclosed. The summary of the interviews suggested that researching information takes longer than it should for them as well as taking notes down in consideration of their fast-paced environment.

Persona:

As explained above, a persona was made to understand the users' needs as well as identify who the relevant users are. For this instance, a persona for a medical staff was created.

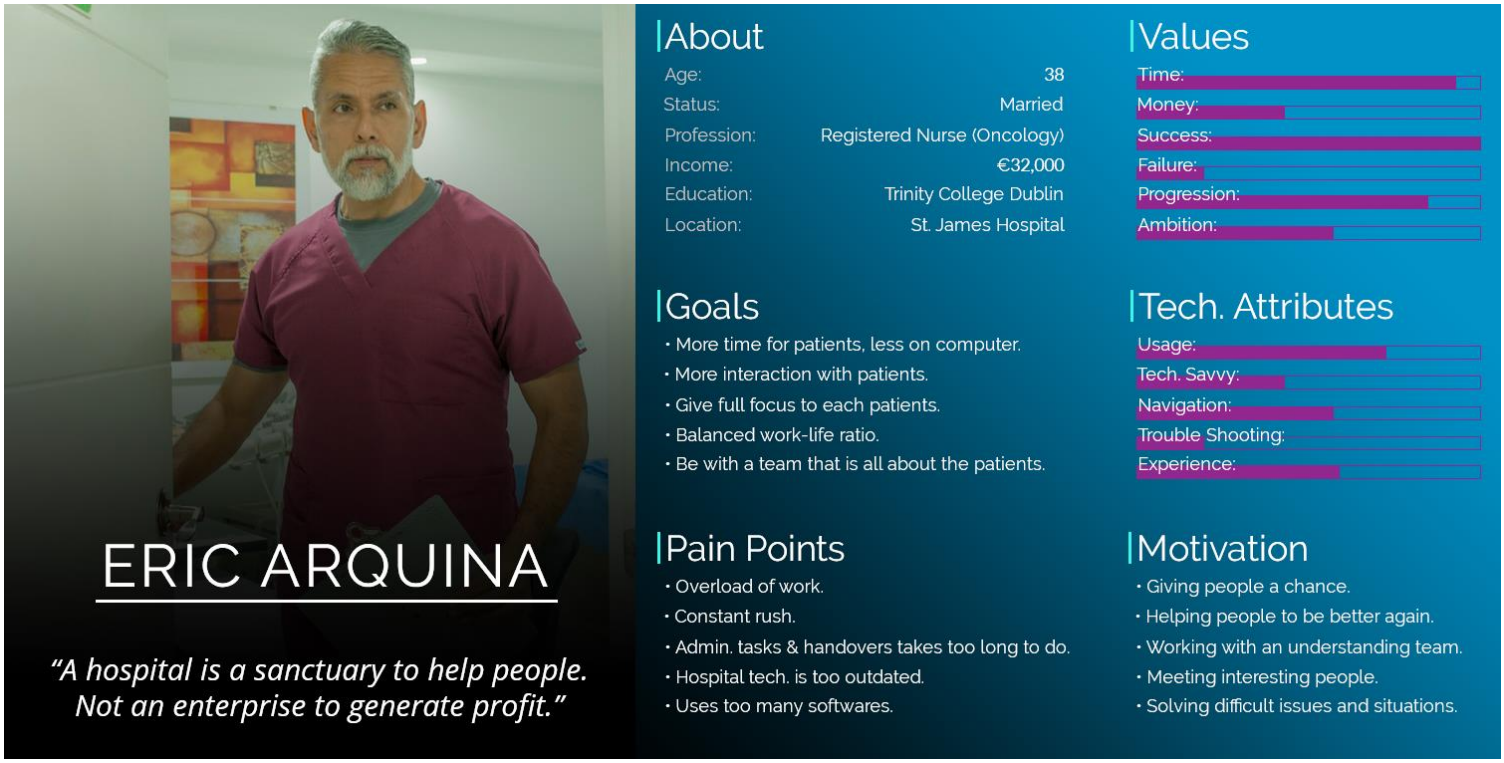


Figure 9 – Persona (Made using Adobe Photoshop CC 2017)

Scenario:

A scenario is a brief fictional story that describes how and why the persona would use the application to complete a particular task in a particular context. The story of this persona is as follows:

Eric Arquina is a RN in the oncology ward. As being a medical staff, there is an extensive amount of information to remember. However, not all of them can be memorised. Due to understaffing, he cares for several patients simultaneously. This means that in a critical situation, he does not have enough time to do the research on a particular symptom.

He requires a tool that he can use inside and outside of work to quickly look up symptoms, its possible causes and its appropriate diagnosis. The tool must be efficient enough to quickly search information in the case of emergency. Should he desire to reference that specific information again, he would like a way to bookmark those as well as the freedom to customise his saved material the way he wants it to be organised. As experienced as John is, there may new data that he has discovered or found an outdated one. He would like a way to publicise his newfound knowledge as well as post his thoughts on an information. This way, he could communicate about a topic with other medical staffs from around the world.

2.3 Requirement Modelling

Requirement modelling is the process used in a project to determine the solutions for the development phase. It defines the requirements of the software application that has been proposed which is seen below. Included in this section are the functional and non-functional requirements as well as a UML – Use Case Diagram. This diagram is used to represent the user's interaction with the system that shows the relationship between them and the various use cases in which the user is involved in.

2.3.1 Functional Requirements

These are the application's feature's or functions that are essential to be implemented to allow the users to accomplish their objectives.

Starting with the most important:

- All users may search information via search bar.
- All users may search information via set categories.
- All users may save (i.e. bookmark/favourite) information pages.
- All users may freely organise saved pages.
- All users may create an account.
- Registered users can CRUD details of their account.
    - i.e. title, first name, last name, profession, hospital (if medical personnel), ward/department (if medical personnel), education and years of experience (if medical personnel).
- All users may read registered users' details.
- Registered users can CRUD their own information pages.
- Registered users can CRUD comments in information pages.

2.3.2 Non-Functional Requirements

These describe how the system must behave and establish constraints of its functionalities. This documents the attributes that of the project that will improve the quality of life for the user when using the application.

Un-ordered list of the application's quality characteristics/attributes:

- Data integrity (i.e. overall accuracy and consistency of data).
- Interoperability (i.e. ability of systems to exchange and make use of information).
- Scalability (i.e. to handle a growing number of users).
- Performance (i.e. response time, throughput, utilisation, etc.).
- Security (i.e. Protection of data).
- Reliability (e.g. server connectivity).
- Maintainability (e.g. debugging application).
- Manageability (e.g. multitasking between unregistered/registered users).
- Usability (e.g. satisfying UX).
- Recoverability (e.g. History list of bookmarked information pages).

2.3.3 Unified Modelling Language (UML) – Use Case Diagram

A Use Case Diagram to depict the possible interactions between the user and the system. Figure 10 and 11 below shows the first iteration of Use Case Diagrams for this project.

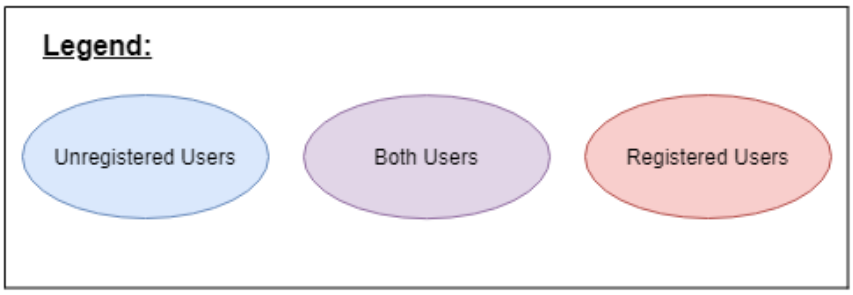


Figure 10 – Use Case Diagram - Legend

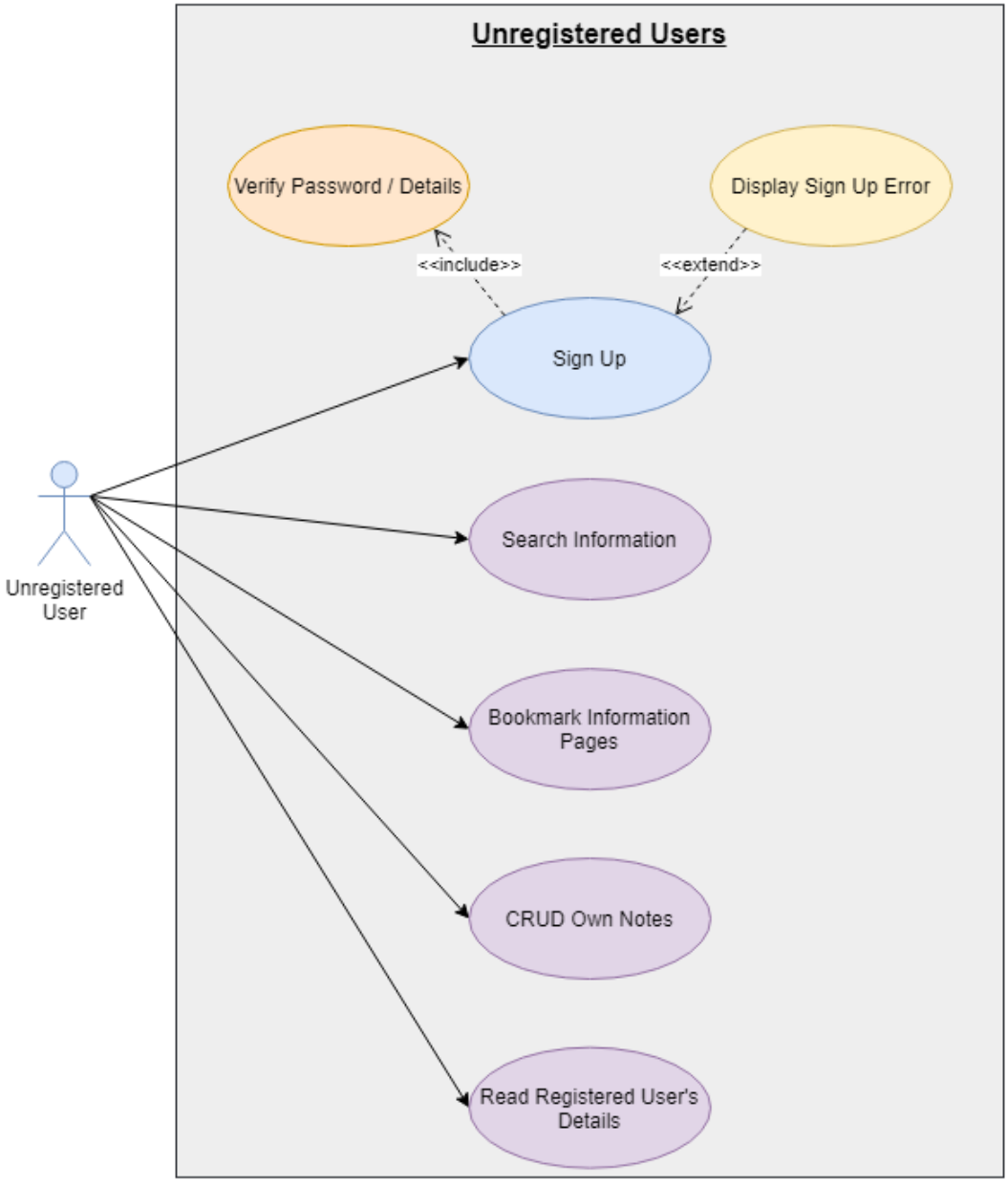


Figure 11 – Use Case Diagram – Unregistered Users

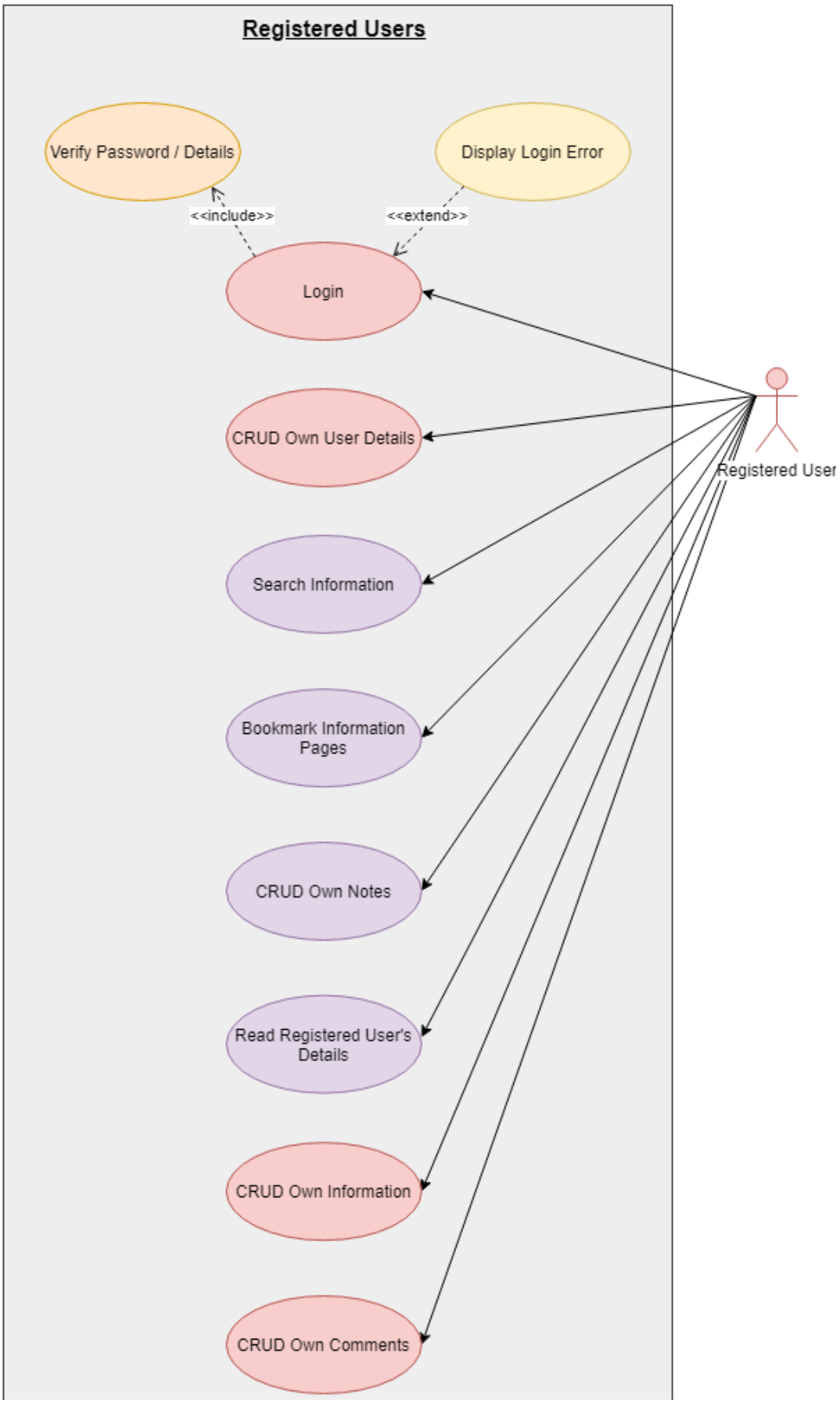


Figure 12 – Use Case Diagram – Registered Users

2.4 System Model & System Requirements

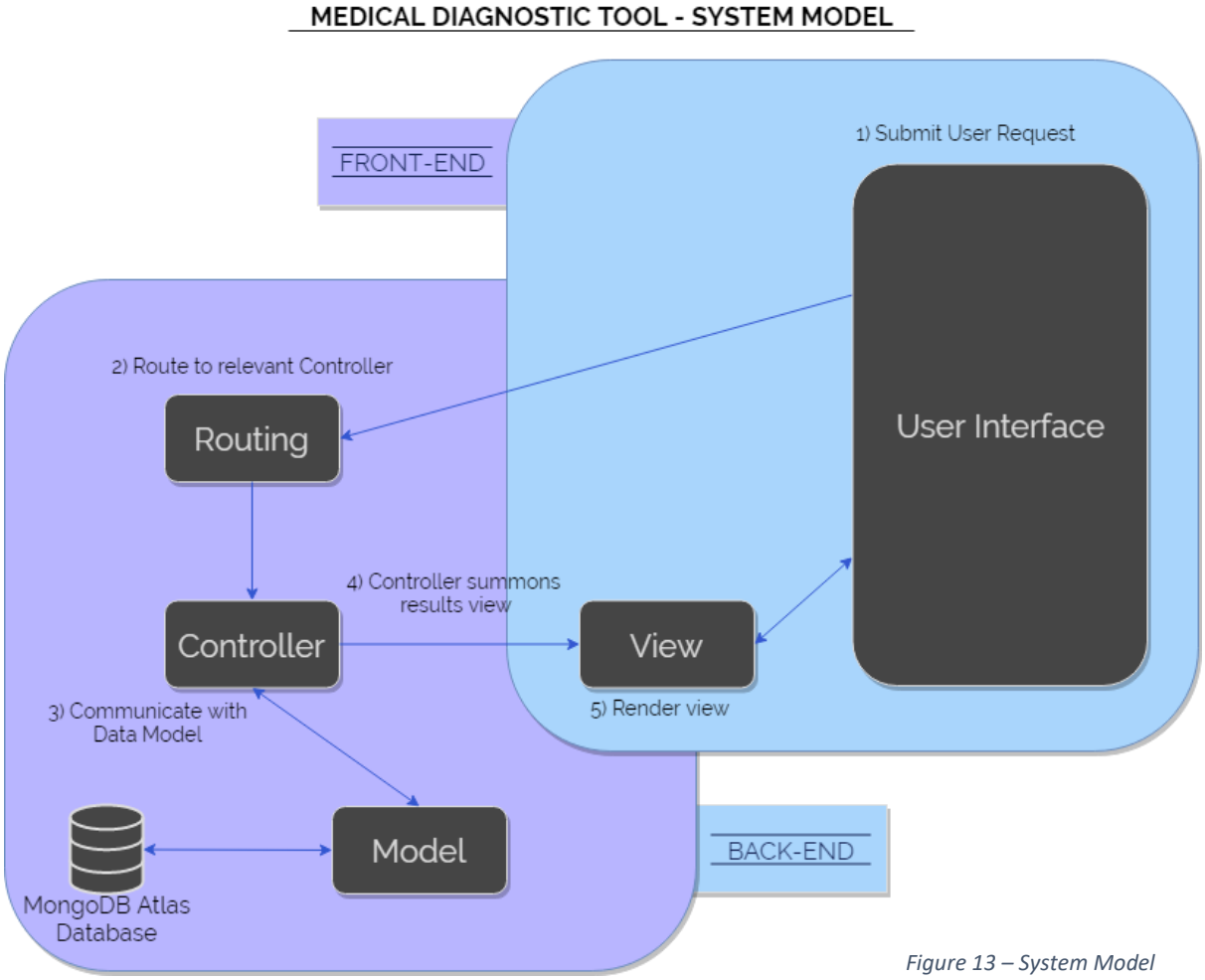


Figure 13 – System Model

This system model (*Figure 13*) follows the Model-View-Controller (MVC) architecture pattern. The Model component communicates to every data-related logic that the user manages. In correlation with the Controller, the Model represents the data being transferred between it. The View component is used for the frontend of the application (i.e. the user interface). While the Controller manages the sent requests, the View displays the results of those requests. The Controller acts as the mediator between the Model and the View. It processes incoming requests, manipulate data using the Model component as well as interacting with the Views to render the results.

The System Model also includes a database and routing component. The database stores the collected data which the Model component may utilise and manipulate. The routing component responds to what page the user goes to. Once the user decides on what to do within the application, the router then corresponds this information to the Controller.

2.4.1 Implementation Platforms

The technologies used for the application are as followed:

- Back-end:
  - MERN Stack
  - Medical Diagnosis API

- Front-end:
  - React.js
  - Atlaskit - Atlassian Component Library

Reasons why these technologies were chosen:

- MERN Stack:
  - A full-stack solution.
  - Allows the developer to easily construct a 3-tier architecture (i.e. front-end, back-end and database) entirely using JavaScript and JSON.
  - High performance to greatly optimise web applications and software.
  - Uses MongoDB.
    - Includes document-oriented storage (i.e. stores data in JSON-like documents) as well as being robust, flexible and scalable.
  - Uses Express.js.
    - A server-side framework that runs on the Node.js platform. It allows for fast development within a Node.js web application, define routes of the application, define an error handling middleware, create a REST API server as well as simplifying the connectivity with databases such as MongoDB.
  - Integrates with React.js.
  - Uses Node.js.
    - A server-side framework that eliminates delays while continuing on with the following request.

- Symptom Checker API:
  - Ease of use.
  - Quick setup.
  - Includes information about symptoms, gender, age possible diseases and diagnosis.
  - Allows the user to search for information mentioned above.
  - Free version includes 100 symptom information per month.

- React.js:
  - Reusable components to significantly save time.
  - One-direction data flow which provides stable codes.
  - Open-source Facebook library.
  - Limitless popular compatible component libraries.

- Atlaskit - Atlassian Component Library:
  - Contemporary design.
  - An abundant number of components in their documents.
  - Supreme customisability.
  - Relatively simple integration with React.js.

Other alternatives that could have been used were Vue.js, Material UI and MySQL. The reason that these technologies were not chosen is due to their lack of features, flexibility and versatility in comparison to their counterpart. While Vue.js is easier to use, has faster performance and includes a smaller project size, it is less robust in comparison to React.js as well as having significantly fewer plugins and component libraries. Its major issues are that some compatible component libraries are complicated to integrate and that it has constant issues with iOS and Safari. Material UI is considered as the best component library alternative in terms of design and integration with React.js. However, the aesthetic and components of Atlassian seems to be more fitting for a medical-themed application. Lastly, while MySQL is simpler in terms of function and integration, similar to the comparison of React.js and Vue.js, MERN Stack has greater flexibility as well as being tailored for the React.js framework.

2.5 Feasibility Study

All four technologies work well with each other. MERN Stack works with React.js, React.js goes together with Atlaskit - Atlassian Component Library and finally the API can be fetched by both the back-end and front-end frameworks.

The main issues that could potentially be encountered are technical issues. This may include challenging bugs, poorly written codes, lack of understanding of documentations or a combination of the above. These issues can be resolved by practising the use of the technologies early on before the development stage as well as by trial and error. During the time of writing this documentation, considering that these technologies have not been used before by the student, understanding these technologies are crucial to eliminate the stated issues as well as decreasing the risk of new ones. This would also help the project management aspect as it would increase the efficiency of the student's productivity.

2.6 Project Plan

The project was designed and developed using six phases. These phases are researching & analysing, outlining the design, further develop the designs, implementation, testing and documentation.

Research and Analysis:

In this iteration, researching the main competitors of this project is necessary to conceptualise features that similar applications do not have. This way, the data gathered through this research may be used to design and develop aspects that has not been used before within the medical research application genre. Researching alternative technologies are also important to either act as backup or supporting platforms (e.g. researching alternative API to either replace the current one or act as a secondary source). Analysing the functional requirements of this project is essential to plan out on what concept feature would be implemented during the development stage as well as what is the most important. Gaining the data from this analysis would also increase efficient workflow throughout the designing and development of the project. A research and analysis of the potential audiences will be needed identify the users; thus, this iteration would include quantitative and qualitative user research methods (i.e. survey, interviews, personas and scenarios) to gather such data.

Outline Design:

This section would consist various draft designs such as the layout of the database (i.e. Entity Related Diagram), flowcharts to represent the workflow of the application as well as wireframes and low-fidelity prototype (i.e. paper prototyping) to visualise the potential UI and UX. It is important to manage the outline designs to aid with the structuring of the project, meaning that these outlines are to be used as the foundation of the application.

Detailed Design:

This will include the updated and definite versions of the outline designs as well as additional documentations. It may contain the finalised version of the ERD, flowcharts, a more detailed version of wireframes, mock-ups, high-fidelity prototyping and a bespoke style guide. These will be used as guides on how the application will look like as well as how it will operate.

Implementation:

The implementation iteration mainly consists of the development stage. This is where the functional and non-functional requirements will be coded using the chosen technologies. This stage is advised to be initiated early on the project timeline as technical issues (as mentioned in *Feasibility Study*) ranging from simple to complicated could arise.

Testing:

This section consists of testing phases to examine the integrity of the application. It may also serve as an influence on either removing, adding and/or updating a current features or designs. Specifically, for this project, it is crucial to test the functional requirements if they are successfully working (e.g. searching up a symptom within the search bar will display the information). It is important to do such testing to identify issues so that the designer and developer may create solutions to eliminate them before submitting the final project.

Documentation Write Up:

Each stage will be documented to thoroughly describe, display and record the history of this project. Reports may include requirements research, design, implementation, testing, project management and business opportunities, all containing in-depth information about this project.

2.7 Test Plan

The test plan consisted of how the project will be tested so that it can be refined as much as possible. The following testing plans are as followed:

Unit Testing:

The importance of unit testing is to obtain a specific reaction or state from a certain function within the application. For this project, the unit testing of fetching information from the medical API would be beneficial to figure out if the fetching process is as accurate and efficient it could be.

Integration Testing:

Also known as Functional Testing, this type of testing is used to examine the behaviour of the user interface interaction. Within this section, the application's navigation, calculation and CRUD system will be examined.

Regression Testing:

This is to examine if a functional and non-functional feature will still perform after a change. For example, should the user decide to bookmark an information then organise it the way they want, the organised information must remain the same once the user changes to a different page.

System Testing:

This is to test the full stack system of the application. For example, once a registered user publicises their own information within the frontend, the back-end must then manage that data then store it within the database. Should the user decide to update that information, the data stored within the database will then be called by the back-end and eventually be displayed in the front-end.

User Testing:

This will consist of the aim of the application, the participant tasks, the details of the participants, information about performing ease of testing, the recommended test environment, test preparations and finally the results from the user testing. The purpose of this is to increase the probability of users utilising this application. To do that, data gathered from this testing is used to improve the UX of the project.

2.8 Requirements and Feasibility Summary

From this chapter, it has displayed the research done for preparation of designing and developing this project. Research done for the competitors, potential users, technologies, functions and plans are used as evidence that this project was feasible to create.

3 Research – Literature Review of Interaction Design

3.1 Introduction

The fundamentals of design thinking are highly valuable to further understand and develop problem solving. At its core, design is a mechanism for developing consequential and innovative solutions that manifest functional and inventive demands. As the inventor of the term “User Experience” stated: “user experience encompasses all aspects of the end-user’s interaction with the company, its services and its products.” (Norman, 1990). In essence, this refers to the true purpose of user experience design which is to prioritise user-centred designing (*What is User Centered Design?*, 2021).

Although User Experience (UX) designing is mainly implemented in the technological sector, this practice may also be applied for all design industries (i.e. product/industrial, web, interior, fashion, etc.). While UX is a design field in itself, it would not be complete without its counterpart, User Interface (UI) design (*The Difference Between UX And UI Design - A Beginner’s Guide*, 2021). Helga Moreno, a senior designer has stated that “something that looks great but is difficult to use is exemplary of great UI and poor UX. While something very usable that looks terrible is exemplary of great UX and poor UI.” (Helga Moreno, n.d). In addition, it has also been said that “UI is the saddle, the stirrups and the reins. UX is the feeling you get being able to ride the horse.” (Dain Miller, n.d).

In this research section, a literature review has been composed to investigate the UX aspect of design, particularly interaction design and user-centred design in terms of the medical diagnostic tool project.

3.2 Interaction Design

3.2.1 Understanding Interaction Design

According to Teo Yu Siang (2020), this sophisticated design category is a term used to describe the interaction between users and products. The purpose of interaction design is to build such products that will allow the user to accomplish their objectives in the most efficient way possible. Elements such as aesthetics, motion, audio, and space are involved to influence the interaction between these two factors. However, each of these elements may also influence specialised fields, for instance, sound design for the composing of audio used in user interaction (Teo Yu Siang, 2020).

In the work of Dan Saffer's Designing for Interaction, he explains that exceptional engineering made devices such as MP3 players to be possible however, it is through interaction design that makes such devices usable, useful and entertaining (Saffer, n.d.). He had also shown examples of the benefits of such design, for instance, buying a product through online shopping. He also examines the result of poor interaction design, for example, struggling to figure out on how to set the clock in a microwave oven (Saffer, n.d.).

He noted that there are three major aspects in terms of defining interaction design. First of which is The Technology-Centred View which states that designers in this field makes digital technology useful, usable and satisfying to use. This aspect also conveys on how these designers uses the raw products generated by engineers and/or programmers and manifest it into products that users enjoy using. Secondly is The Behaviourist View which examines the focus on how products behave as well as how it provides feedback based on how users operate with them. Lastly is The Social Interaction Design View which revolves around assisting the communication between users through products. This broad aspect may take various forms such as the ability to be one-to-one (e.g. telephone calls), one-top-many (e.g. a blog) or many-to-many (e.g. stock market).

The meaning of interaction design has been summarised that when a behaviour of a product is involved, for the best experience possible, an interaction designer would then also be involved. This claim is supported by Saffer's The Disciplines of User Experience Design Venn diagram (*Figure 14 – Saffer's Venn Diagram*).

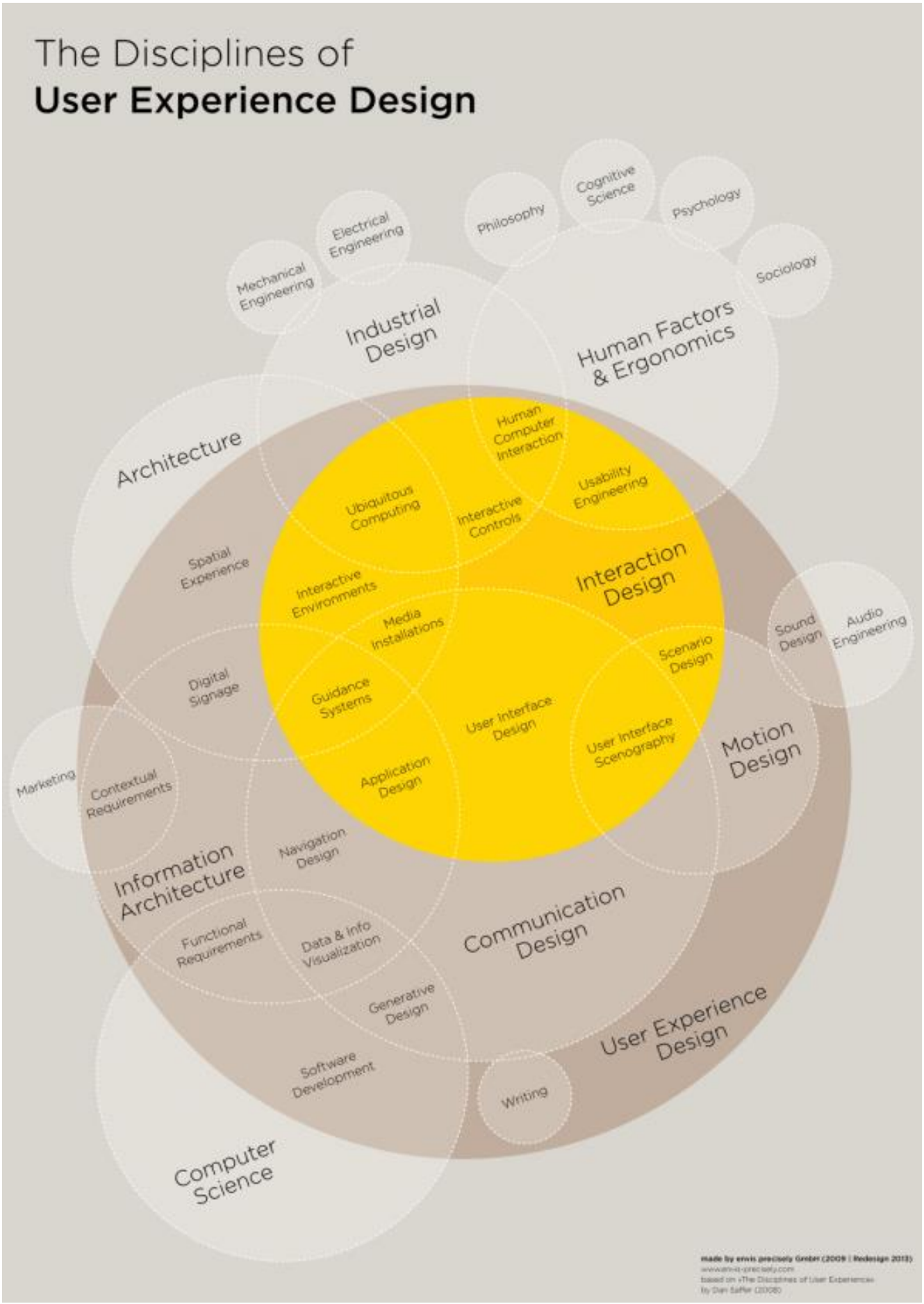


Figure 14 – Saffer's Venn Diagram **(Wilson, 2013)**

3.2.2 Approaches that Interaction Design Employs

Included in Dan Saffer's research paper are interpretations of seven approaches on how and why interaction design is applied:

Focusing on Users:

This approach defines that the users does not understand (let alone does not want to) on how a corporation is operated and/or structured but only desire to achieve their goals by doing their tasks. This concept also defines designers as the advocates for end users (Saffer, n.d.).

Discovering Alternatives:

This approach explains that designing is not about deciding among multiple alternatives, but it is about creating options until the latest option is the most desirable one among the rest. This method of making multiple alternatives and choosing the latter is claimed to set designers apart. To further understand this approach, Saffer had included an example of which he demonstrated by Google's AdWords' situation; The company required public relations to generate revenue via advertisement however, the users were displeased of the traditional banner placements thus leading to the creation of text advertisements (Saffer, n.d.).

Utilising Ideation and Prototyping:

This approach explores the finding of solutions via brainstorming, prototyping and then by testing. Saffer announced that in comparison to how the prototyping method from various fields (i.e. science, engineering, business, etc.) are similar with different traits in their own rights, within the design field, it is substantially different. This claim is supported by evidence stating that design prototypes are not fixed, meaning that any specific prototype does not entirely produce a solid solution. To further understand, this also means that the use of several prototypes may be used to manifest a single product (Saffer, n.d.).

Collaborating & Addressing Constraints:

This approach features the topic of designers working as a team instead of working alone. It also describes adds that while designers dream of ambitious concepts, these ideas comes with constraints such as addressing business goals, compromising with teammates and meeting deadlines, hence the emphasis on team effort in terms of designing (Saffer, n.d.).

Developing Appropriate Solutions:

This approach signifies designers carrying their experience and wisdom throughout their upcoming project. While this circumstance may be beneficial for future works, the preeminent solution should uniquely focus on the issues for the specific problem of that specific project. Saffer had recognise that this kind of problem-solving method should not be taken religiously, however, using the same solution in other contexts cannot or should not be identically mimicked for other projects. An example of this would be Amazon; While this company has an outstanding e-commerce model, it cannot be completely cloned elsewhere (Saffer, n.d.). Should a team of designers have been tasked to create an e-commerce prototype, design solutions will have to be appropriate to its situation.

Drawing on a Wide Range of Influences:

This approach briefly describes on the influence of design has on various subject fields (i.e. psychology, ergonomics, engineering, architecture, art, etc.) and how designers delivers an extensive multidisciplinary spectrum of concepts from which to attract inspirations as well as solutions (Saffer, n.d.).

Incorporating Emotion:

This last approach analyses that, in terms of analytical thinking, emotions are perceived as an obstruction to logic as well and making the appropriate choices. However, in design, emotion is required to be included within decisions. Saffer has included an example to support this statement by noting ‘what would the Volkswagen Beetle be without whimsy?’ (Saffer, n.d.).

3.2.3 The Five Dimensions of Interaction Design

The five dimensions model is created by Gillian Crampton and Kevin Silver (Teo Yu Siang, 2020) and is used to understand what this aspect of UX design involves.

First Dimension – Words:

Particularly the ones utilised in interaction (e.g. button labels), it should communicate information to the user however, not so much that it would overwhelm them thus, the designing of this should be minimalistic while making it meaningful and simple to understand (Teo Yu Siang, 2020).

Second Dimension – Visual Representation:

This contains graphical elements such as images, typography and icons that users may interact with. These elements commonly correspond with the words used to address information to users (Teo Yu Siang, 2020).

Third Dimension – Physical Objects or Space:

The third dimension explains that everything and anything will affect the interaction between the user and the product. A few examples have been said by Teo Yu Siang to support this claim. Those examples examine on what physical objects users interacts with the product such as a touchpad for a laptop or the user's fingers for smartphones. It also depicts examples on what type of physical space the user is in when they interact with products such as standing in a crowded environment while using their smartphone or within their office space surfing the web (Teo Yu Siang, 2020).

Fourth Dimension – Time:

This dimension mainly refers to the media's changes with pace (i.e. animation, videos, audio and progression in general). Utilising the user's sense of sight and hearing plays a critical factor in offering visual and audio feedback to their interactions. Progression is also regarded within time. For instance, the amount of time the user spends interacting with the product (Teo Yu Siang, 2020).

Fifth Dimension – Behaviour:

This involves the execution of a product meaning, how the product dictates the user's operation of the product as well as how they would perform actions (Teo Yu Siang, 2020).

3.2.4 Human-Computer Interaction (HCI)

Within interaction design is the human-computer interaction (HCI) multidisciplinary field. This studies the focus on the design of computer technology, specifically the interaction between the user and the device. While HCI originally focuses on computers, it has since broadened to meet most practices of information technology design (*What is Human-Computer Interaction (HCI)?*, 2014) as demonstrated in *Figure 15 – HCI Diagram*.

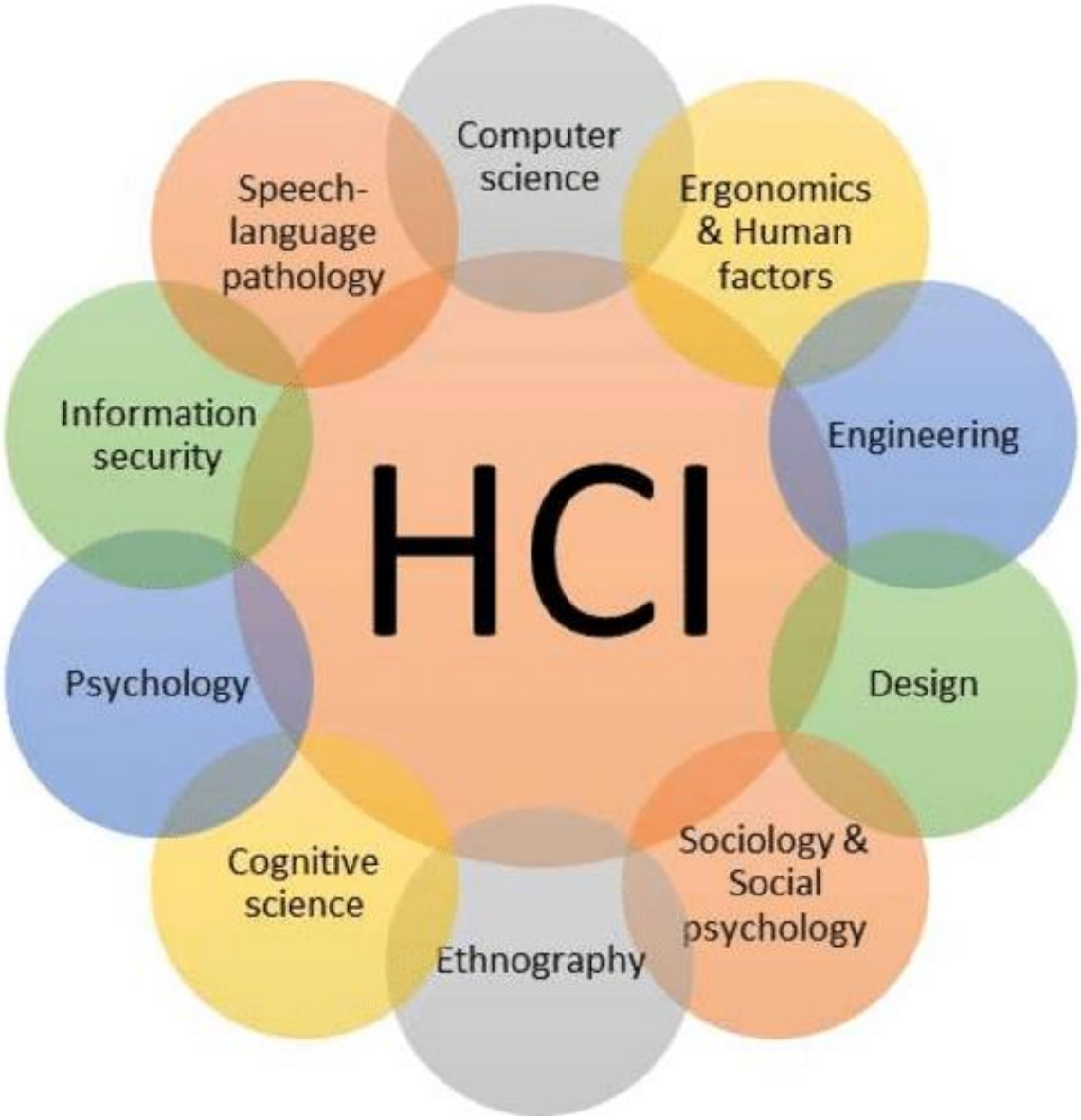


Figure 15 – HCI Diagram ***(Institution of Engineering and Technology, 2017)***

According to Gerard Jounghyun Kim in *Human-Computer Interaction Fundamentals and Practice*, the HCI has become a major importance in terms of computer design as such devices has been a household name in most aspects of peoples' lives (Gerard Jounghyun Kim, 2015). In other words, technology has become second nature for us.

It has been explained that the early - and still present - focus of HCI is the prioritisation of high usability in consideration of design interaction and interface implementation. The definition of the term "high usability" refers to the resulting interfaces being simple to use, highly efficient for the objectives, provides safety, and accompanies the users to the appropriate completion of the objective.

The combination of minimalistic aesthetics and satisfying the requirements for usability has been mentioned to be increasingly necessary for commercial success. An example has been provided in the form of the Apple company. Due to their products being critically acclaimed for their attractive design, this has generated the enterprise an exceptional number of loyal followers even though that the functionalities of their hardware and software may be similar - if not slightly inferior – in comparison to some of their competitors (e.g. Microsoft, Samsung, etc.).

3.2.5 The Principles of HCI

The level of difficulty of generating good HCI design is claimed to be complicated due to the multi-objective assignments that involves simultaneous consideration of various factors (Gerard Jounghyun Kim, 2015). These factors include (but not limited to) the users, characteristics of tasks, capabilities and expense of the products, deficiency of objectives and the evolution of technologies. To decrease the difficulty of making good HCI design, Gerard Jounghyun Kim has said that multiple researchers and developers have created and published fundamental principles in the hopes of accomplishing such objective (Gerard Jounghyun Kim, 2015). These principles are claimed to be applicable to most HCI design circumstances.

"Know Thy User":

Similar to Dan Saffer's *focusing on users'* approach, this principle states that the interaction, as well as the interface, should provide to the requirements and capabilities of the users. A sample of the process for this principle is by collecting comprehensive information (also known as quantitative data). These data may include the age, gender, education level, social status, technological experience, cultural background and etcetera as an example. The information gathered will then be analysed to decide the users' possible preferences, tendencies, capabilities and skill level (Gerard Jounghyun Kim, 2015).

Understand the Task:

This principle is exactly what the name describes; the focus is to comprehend the task involved. Understanding the task at hand is relative to the modelling of interaction as well as user analysis. However, it has been emphasised that the task model must specifically occur from the user/s (Gerard Jounghyun Kim, 2015).

Reduce Memory Load:

The meaning of "memory-load" does not relate in terms of computer architecture however, it relates to the users' capability of holding a certain amount of information within their own memory. This principle explains on how the performance of humans depends on the quantity of memory burden. For instance, for people to achieve peak efficiency for performing tasks, those said tasks requires less memory burden (Gerard Jounghyun Kim, 2015).

Reminding Users & Refreshing their Memory:

As stated in the previous principle, the users' memories must be recognised to efficiently design an HCI. In this principle, it has been said that any major tasks will most likely involve the use of memory, thus an alternative strategy is to make use of interfaces that provides consistent reminders of critical data and thereby refreshes the users' memories. This concept is crucial to recognise due to how the human memory dissipates information quickly, as evident to when switching tasks in a multitasking environment may confuse the person, thus will result in an inefficient performance (Gerard Jounghyun Kim, 2015).

Preventing Errors/Reversal of Action:

This principle consists of the importance of designing an error-free operation for the users. It is considered as crucial to designing a quick completion system for the tasks. It has been said that the interaction and interface should be designed in a way to prevent confusion and mental overload (Gerard Jounghyun Kim, 2015).

Naturalness:

This final principle emphasises on the favour of natural interaction and interfaces, meaning, the reference to traits that is reflective of many activities in the daily lives of people. An included example is that flawless HCI may soon be realised when a natural language-based conversational interface is plausible (Gerard Jounghyun Kim, 2015).

3.3 User-Centred Design

One of the major aspects of interaction design is user-centred design which, according to the *Interaction Design Foundation*, is an iterative design process where designers prioritise on the users and their requirements (*What is User Centered Design?*, 2020). This refers to a team of designers associating users during the whole design process by conducting various research and design methods to manifest a usable and accessible product/s for them.

Each iteration of user-centred design provides four unique phases. These phases include designers collaborating as a team to understand the context in which users may utilise the product's system. The team would then identify and target the users' requirements. After the previous follows a design phase in which the team conceptualises multiple solutions. The next and final phase involves the evaluation as well as the assessment of those evaluations against the users' context and needs. This purpose is to check the efficiency of the design's performance. Should the four phases be accomplished once, the team would then require to further iterate on these four phases until satisfying results from the evaluation is obtained (*What is User Centered Design?*, 2020). Further in-depth details of such phases are listed within the next page:

3.3.1 The Four Phases of User-Centred Design

Analysis Phase:

According to Paul Rouke from the *What is User Centred Design* article, this phase assures that all business and user needs are taken into consideration before the starting the design phase (*What is User Centred Design? A guide to the processes involved | PRWD*, 2017). Methods included within this phase are stakeholder analysis, competitor benchmarking, persona and scenario development, attending field studies and finally defining usability objectives.

Design Phase:

This phase ensures an inclusive procedure to the design of the product, which would then ensure the design to accomplish all business and user needs. Methods includes the development of navigation models, screen flow diagrams, information architecture, card sorting, paper (low-fidelity) prototyping, wireframing, interaction design and user testing (*What is User Centred Design? A guide to the processes involved | PRWD*, 2017).

Implementation Phase:

This phase acquires the results of the user tested system and interaction designs then integrates them into build-ready systems that is ready to go live. Methods includes object-oriented design, user interface (UI) implementation, server implementation, heuristic evaluations, further user testing and finally documentation (*What is User Centred Design? A guide to the processes involved | PRWD*, 2017).

Deployment Phase:

This final phase involves consistent evaluation of the latest system of the product, meaning, the constant monitoring of the performance against usability objectives. Methods included are user surveys, further field studies, performance analysis, improvement scoping and continuous analysis (*What is User Centred Design? A guide to the processes involved | PRWD*, 2017).

3.3.2 Human-Centred Design in Medical Technology

In the article *Human-centred design in hospital technology* composed by Paul Le (2020), he explains that designers are unfortunate to be taken for granted within the technology field which results in displeasuring consequences, particularly in the healthcare IT industry. As an example, Le had clarified that a crudely-designed UI will result in wrong information being submitted by a medical staff into an electronic system (2020). In this autobiographical article, Le describes the methodologies he utilised to build something that solved true problems, instead of building something that was aimed on the technology being used.

Discovering:

This methodology helps the designer to define and understand the issue by narrowing down the focus. From his user research through various medical staff, Le gained data on the situation of the medical workforce against technology. Such information where healthcare workers that he has researched are not as tech savvy. Another finding from the research was that many hospitals are presently progressing towards integrating modern technology for relaying critical information. Finally, the issue that nurses have in regard to efficiently improve decision-making process within the ER department. For instance, there is a requirement for improved organisational and efficiency of managing incoming medical information. The insight gained from this user research was the opportunity to determine a crucial issue within emergency rooms, particularly since hospitals that Le had researched are moving towards implementing modern technology into their workflows. Through this process, Le has demonstrated the importance of discovering the issue of a topic, in this case, it would be improving the decision-making process within the ER department (2020).

Brainstorming:

The purpose of the brainstorming phase was to identify solutions to permit medical staff to quickly obtain critical medical information for patients that are delivered to the ER. Another purpose was to improve the method that patient information is displayed to medical staff, thus creating a solution for them to make it simpler to analyse the information particularly during a stressful situation (Le, 2020).

Prototyping:

The last methodology that Paul Le used was the development of prototypes. This is to ensure that ideas and solutions are communicated to the problems using insights gained from the *discovering* and *brainstorming* process. Similar to Dan Saffer's *developing appropriate solutions* approach, prototypes are created until the latest one results the most satisfying outcome, thus being an iterative process. This includes defining user stories, wireframing, Lo-Fi and Hi-Fi prototyping then finally user testing. Through this, Le was able to design a feature that will allow users to search for patient information in a database as well as displaying such information in a UI. Additional prototypes were the function that permits the user to input information as well as a rating feature in terms of the patient's severity (Le, 2020).

3.4 Research Summary

The work of Teo Yu Siang's, Dan Saffer's, Gerard Jounghyun Kim's and Paul Le's literature as well as various articles has been acknowledged in this chapter. By considering those researches, the exploration through interaction design and its aspects has resulted to the understanding of its sophistication and its necessity within UX design. Summarising this literature review, it is evident that the focus of interaction design is to cater for the needs for users by thoughtfully regarding the approaches, the five dimensions, the HCI and the four phases of user-centred design. By doing so, the designer may produce a product/s that will be user-friendly and provide a positive user experience. This research will be referenced in the research and design of the medical diagnostic tool to inform the user experience and user interface design of the project.

4 Design

4.1 Introduction

The application allows medical staff and medical students to quickly search information about a symptom as well as permitting the diagnostics and investigation of the cause of such symptom. Furthermore, the application allows the user to save the information that they have viewed as bookmarks and create quick notes for them to review. This project is designed to run seamlessly on mobile devices, meaning that this project will be responsive in any end-devices (i.e. smartphones, tablets, laptops, etc.). This project is also designed to be user friendly, meaning the UI of the application is focused on ease of accessibility.

4.2 Program Design

Program design consists of phases the developer should complete before initialising the coding. Should these phases be documented appropriately, it will allow other developers to maintain the completed program in the future. The program design will explain the technologies that has been used, the arrangement of the MERN Stack, the design patterns, the architecture of the application and finally the database and process designs.

4.2.1 Technologies

The technologies used for this project as discussed in the *Requirements & Feasibility* chapter largely remained the same with minor updates.

The project used a MERN Stack application with the current MongoDB Atlas database however, the API was updated from the *Medical Diagnosis* API to the *System Checker* API.

The front-end framework is React.js however, the component library was updated from *Atlaskit – Atlassian* to *Google Material UI* design system. This is due to word stating that *Atlaskit* will deprecate some of their UI components which - as of writing this documentation - has been found within their component guidelines (*Atlaskit by Atlassian*, 2021). While not a major issue to some and it may be possible to work around it, the instability of this has been deemed to increase the chance of compromising the project.

The second reason that *Atlassian* has been replaced is due to its complexity to integrate their component libraries to the project. While the component library is not at fault for this reason, it is due to the programming capability of the student.

The third and final reason is that *Material UI* has all of the components that is needed to develop the application like how it was designed to be. While not as medically-aesthetic in comparison to *Atlassian*, it does have the features to develop on what was planned on the wireframes, mockups and high-fidelity prototypes.

4.2.2 Structure of MERN Stack

This full-stack development method is constructed after the key technologies; MongoDB, Express.js, React.js and Node.js, hence the abbreviation *MERN*. This is one of several variations of a full-stack solution which others consists of *MEAN* (Angular.js) and *MEVN* (Vue.js).

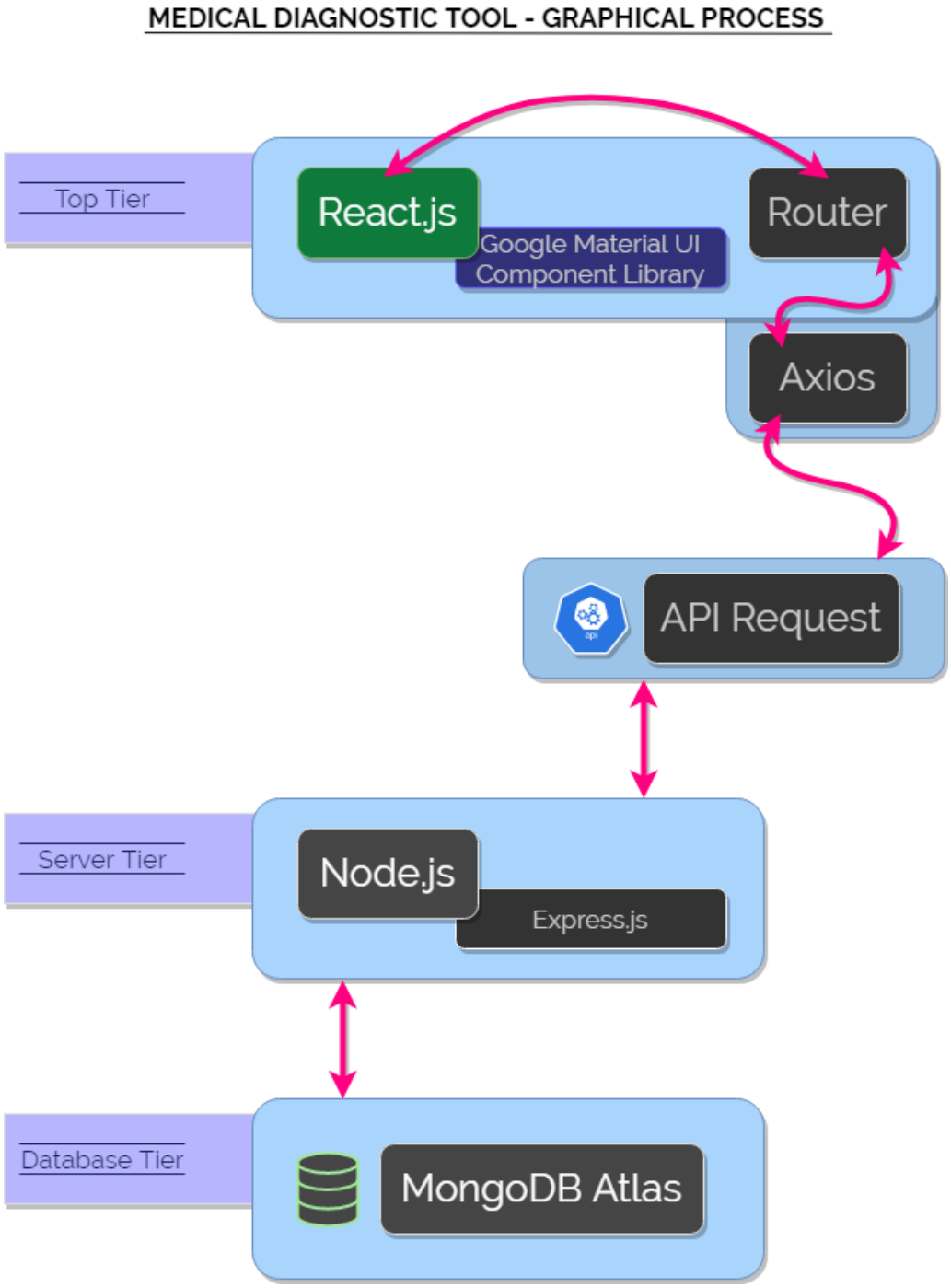


Figure 26 - Structure of MERN Stack

MERN Stack allows the developer to build a 3-tier architecture using but JavaScript and JSON only without difficulty. The three tiers as displayed in Figure 16 are the Top Tier, Server Tier and the Database Tier.

Top Tier:

The top tier consists of the frontend design. It includes React.js which is a declarative JavaScript framework for developing dynamic client-side applications in HTML. This frontend framework allows the developer to construct sophisticated user interfaces by utilising components, connect them with the server tier and render them as HTML. Included within this tier is also the component library which is used in conjunction with React.js to display the UI and the routing functionality. The router is connected via *Axios* (a library that permits developers to execute HTTP requests to external resources) to allow the client-side to fetch, update or remove data from the API.

Server Tier:

The tier below the client-side contains the Express.js server-side framework which runs inside the Node.js server. Express.js is considered to be as a quick, arbitrary and a minimalist web framework for Node.js. This is evident from its powerful models for URL routing (i.e. matching an incoming URL with a server function) as well as how it handles HTTP requests and responses. Using Express.js to power the application will then use MongoDB's Node drivers via callbacks for using Promises as well as to access and update data within the database itself.

Database Tier:

Should the application require to store any data (e.g. users, comments, articles, contents, etc.), it is advisable to store the data in a database. In terms of the Medical Diagnostic Tool, the database chosen is MongoDB Atlas, a cloud-based web version of the original MongoDB software. JSON documents created within React's frontend may be sent to the Express.js server where they can be processed and stored directly in the MongoDB Atlas database for later use.

4.2.3 Design Patterns

Design patterns are solutions for solving development problems in an efficient way as much as possible. They act as guidelines for developers on how to develop the application by offering them a variety of approaches. The patterns used for this application is the Model View Controller (MVC), REST API and React's own design patterns.

Model View Controller (MVC):

In React, this pattern is used to separate application's concerns. The Model represents an object bearing data, the View represents the depiction of the data that the Model contains, and the Controller governs the data flow into the Model object and also updates the View whenever data is altered.

REST API:

REST API is an abbreviation for Representational State Transfer Application Program Interface. This is based on an architectural style and approach to communications mainly implemented in web services development. This design pattern uses HTTP requests to GET, PUT, POST and DELETE data.

React's Design Patterns:

There is a large amount of design patterns that can utilised with React. the three patterns that are discussed below are recommended for developers that are novices to the structure of this framework (Jones, 2019).

- Controlled Components:

This pattern describes how React manages the updates of the DOM for the developer within a component. In a controlled component, the framework administers the states. Unless there are legacy JavaScript codes that will be used and is not written for React, using an Uncontrolled Component is advised. Otherwise, it is recommended to always use controlled components (Jones, 2019).

- Stateless Functions:

Stateless functions are also known as vanilla JavaScript functions. This is used for when simple components are being built within a React application. By utilising this design pattern, it is not required to create classes which extends *React.Component* which will provide multiple backend state managements thus making the application run quicker. However, should the developer decide to build a simple component, extending the class is deemed to be unnecessary. As the title of pattern may suggest, this describes a component without states and event handlers. Once a stateless function has been created, only props may be passed into the render function.

- Conditional Rendering:

This design pattern is similar to the concept of if() and else() statements however, written in a JSX format. An example of this is seen in Figure 17:

```
{user === null &&
  <Nav.Item>
    <Nav.Link as={Link} to="/register" className="tertiaryCol">
      Register
    </Nav.Link>
  </Nav.Item>
}
```

Figure 17 - Conditional Rendering Example

This example depicts that if the user is not logged in, display the Register link within the navigation bar.

4.2.4 Application Architecture

Application architecture refers to the patterns and methods used to design and construct an application. It provides the developer a roadmap and recommended approaches to follow when building an application. This will aid the developer to create a well-structured coding project.

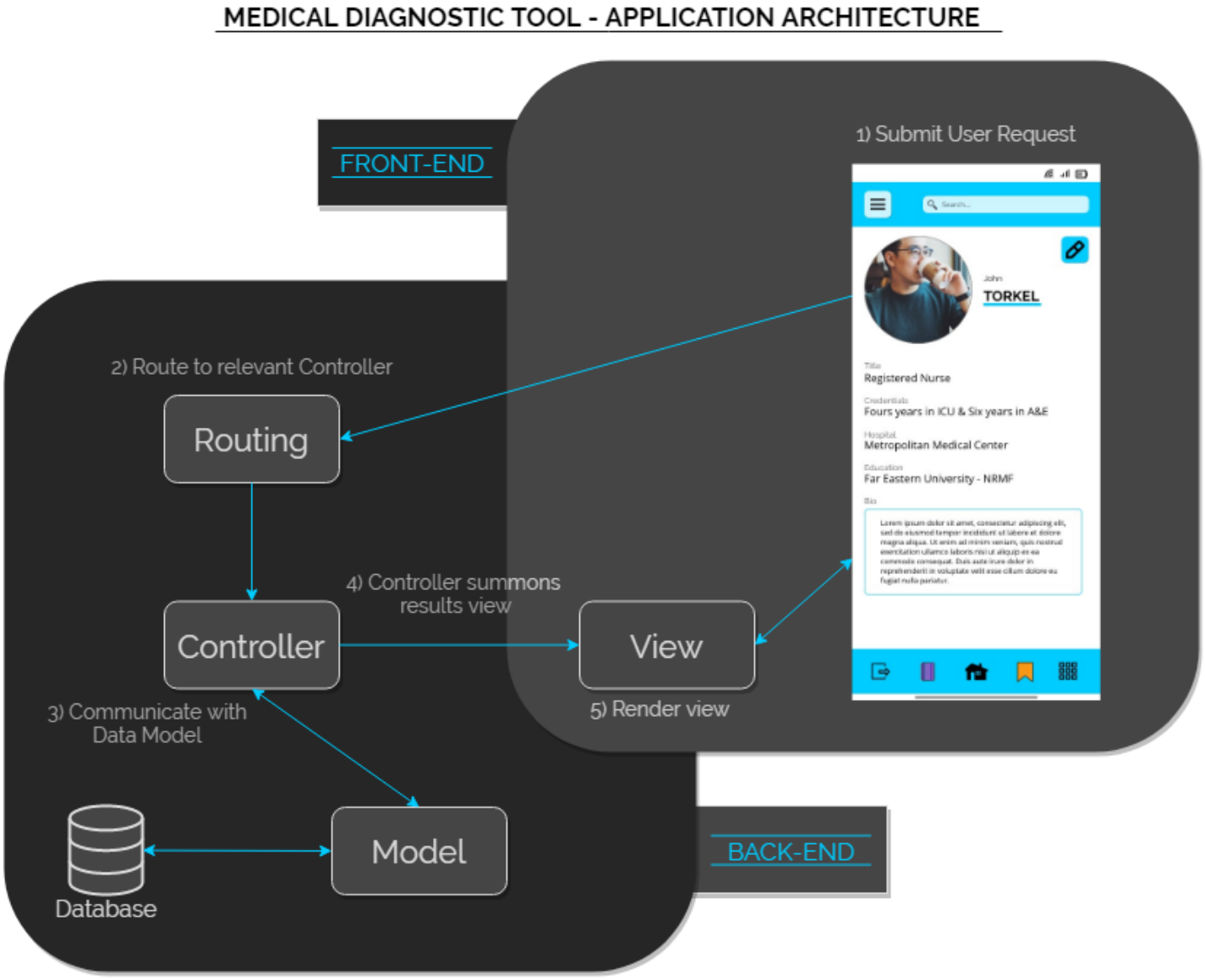


Figure 18 - Application Architecture

**Based on Mohammed Cherbatji/Web Application Framework/Slides/1 - Laravel and MVC*

4.2.5 Database Design

An Entity Relationship Diagram (ERD) is a structural blueprint that contains database-related symbols and connectors to visualise the major entities within the system scope as well as the inter-relationships among those entities. Should the application require to store and retrieve data for later use, a database is a necessary integral part for the software system. An ERD represents those data that the application will manipulate.

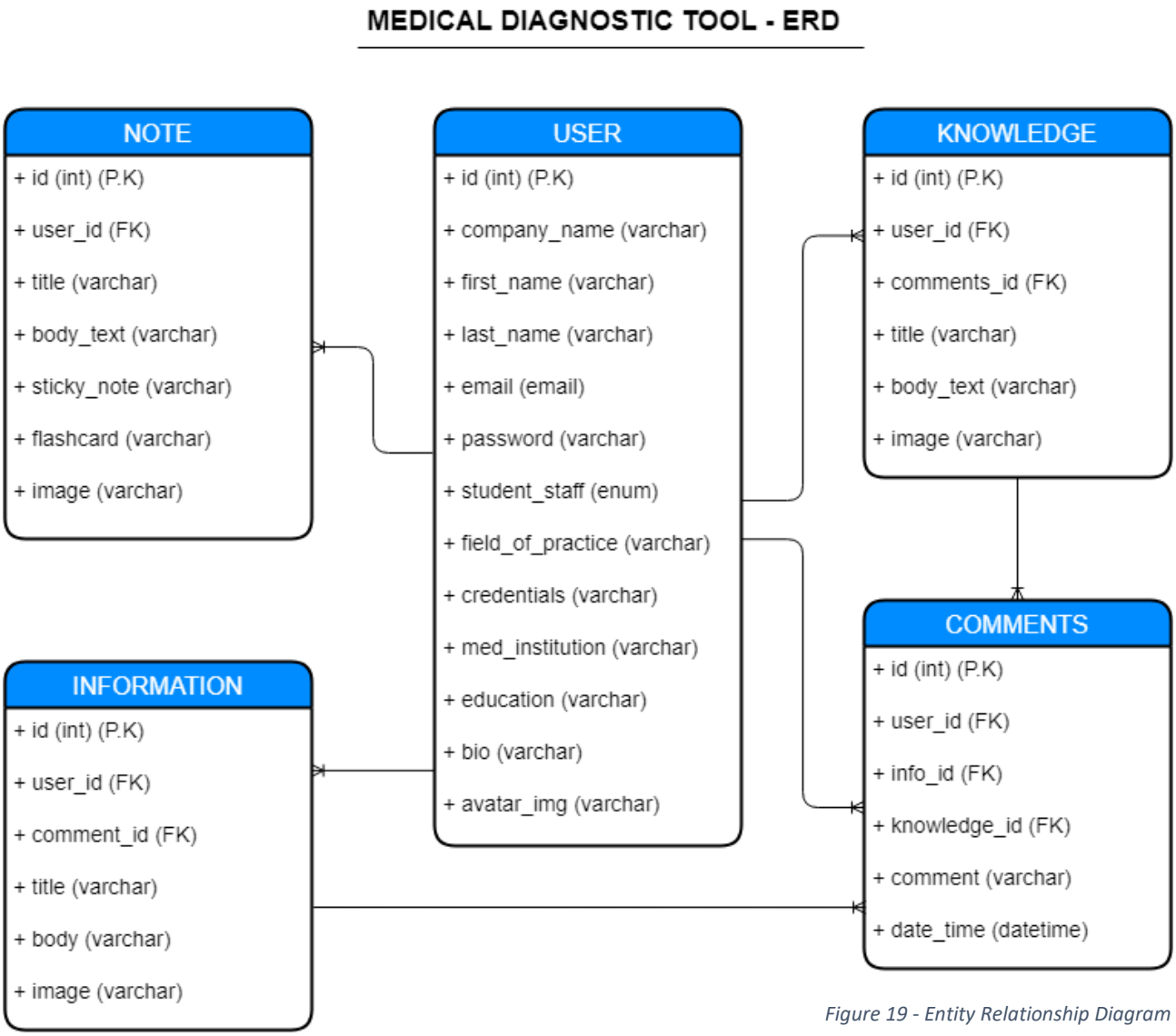


Figure 19 - Entity Relationship Diagram

4.2.6 Process Design

A flowchart represents the visual process of the application. Should there be any sophisticated documents, study and/or plan for the structure of an application, a flowchart is used to simplify the explanations of the process for the software.

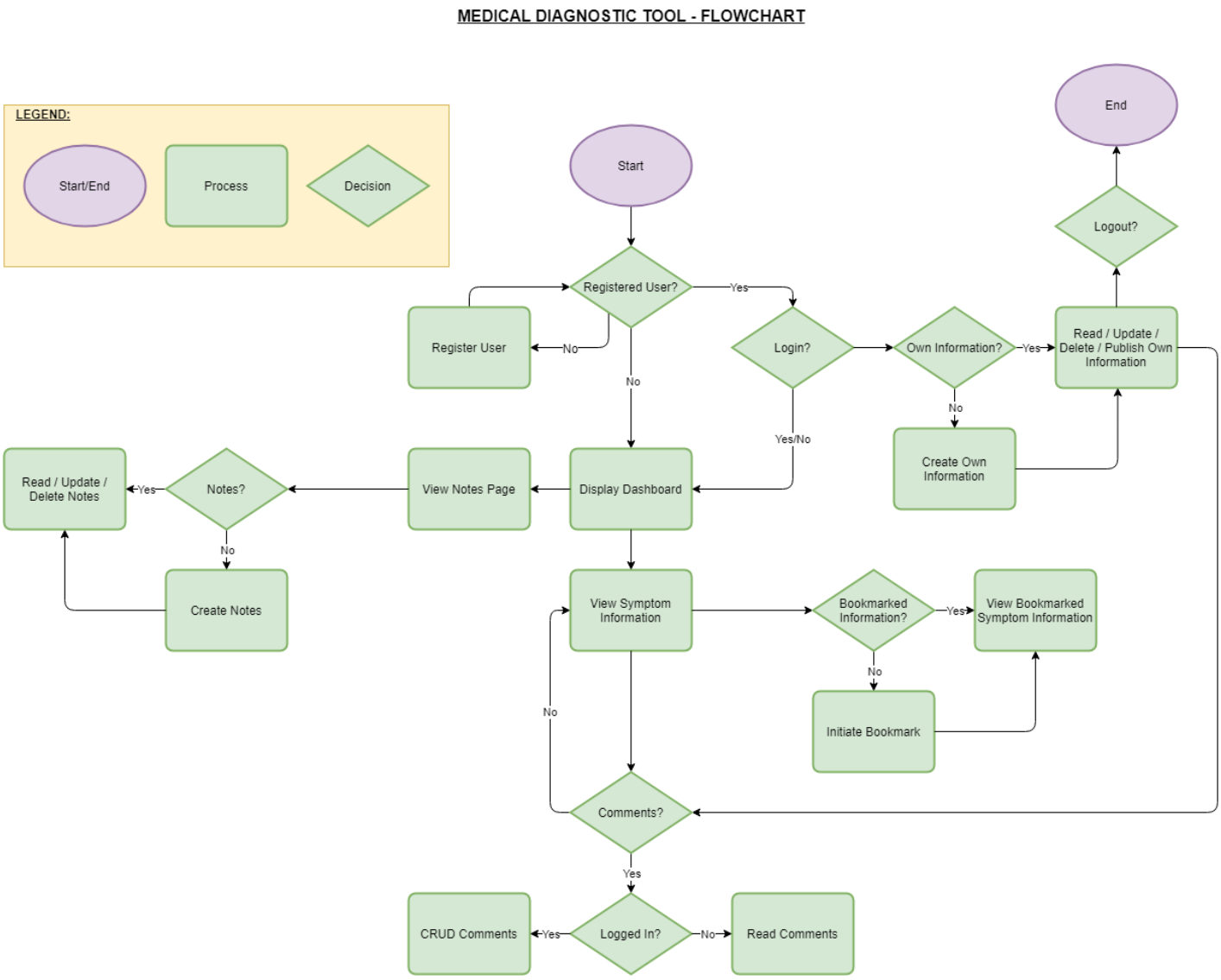


Figure 20 - Flowchart (Iteration 1)

4.3 User Interface (UI) Design

While its counterpart (user experience design) is known to be the "feeling" of the product, user interface design is known as the aesthetics of it (*The Difference Between UX And UI Design - A Beginner's Guide*, 2021). The UI of an application is just as important as its UX as the look of the application may potentially attract additional users to operate the product. A combination of both good UI and UX is a crucial factor to develop an application that users will appreciate. While UX design subconsciously navigates the user through the application, UI design allows the user to interact with it via well placed, appropriately sized, colour-coded and with aesthetically pleasing components.

4.3.1 User Flow Diagram

A user flow diagram examines on how a user will navigate through a product. It arranges the user's movement through the product while mapping each and every step the user will potentially make starting from the entry point all the way to the final interaction. Similar to the flowchart, should there be any sophisticated documents, study and/or plan for the structure of the UI, a user flow diagram is used to simplify the explanations of the process for the user's navigation through the product.

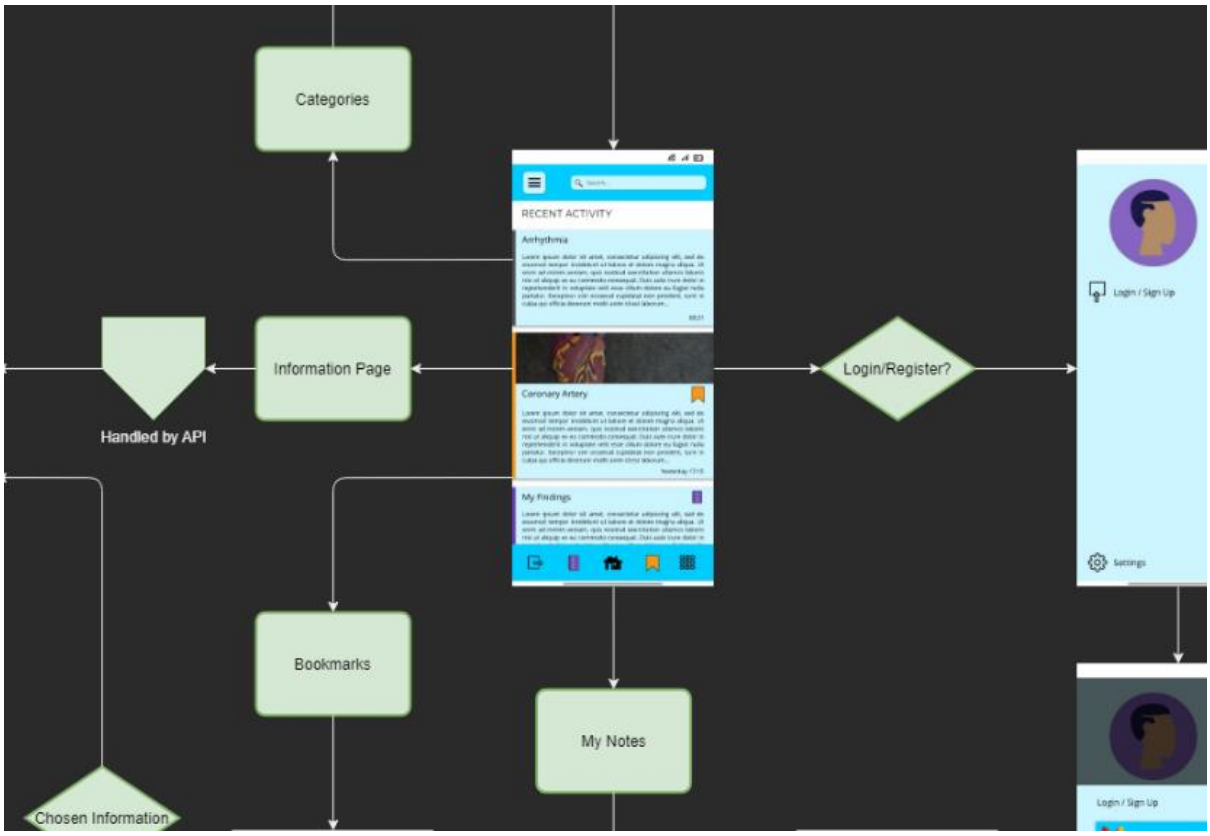


Figure 21 - User Flow Diagram (Iteration 1). See Appendix D – Diagrams for full User Flow

(Images used are made using Invision Studio)

4.3.2 Style Guide

The importance of a style guide is that it provides designers with a sense of direction and little room to stray beyond the design principles. In terms of the career sector, style guides are made by a designer or a design team and is utilised as the primary communication tool between production teams and clients. It ensures a consistent brand experience, meaning, no matter how and where a customer experiences a product, the customer will be experiencing the same fundamental design attributes.

Chromatics

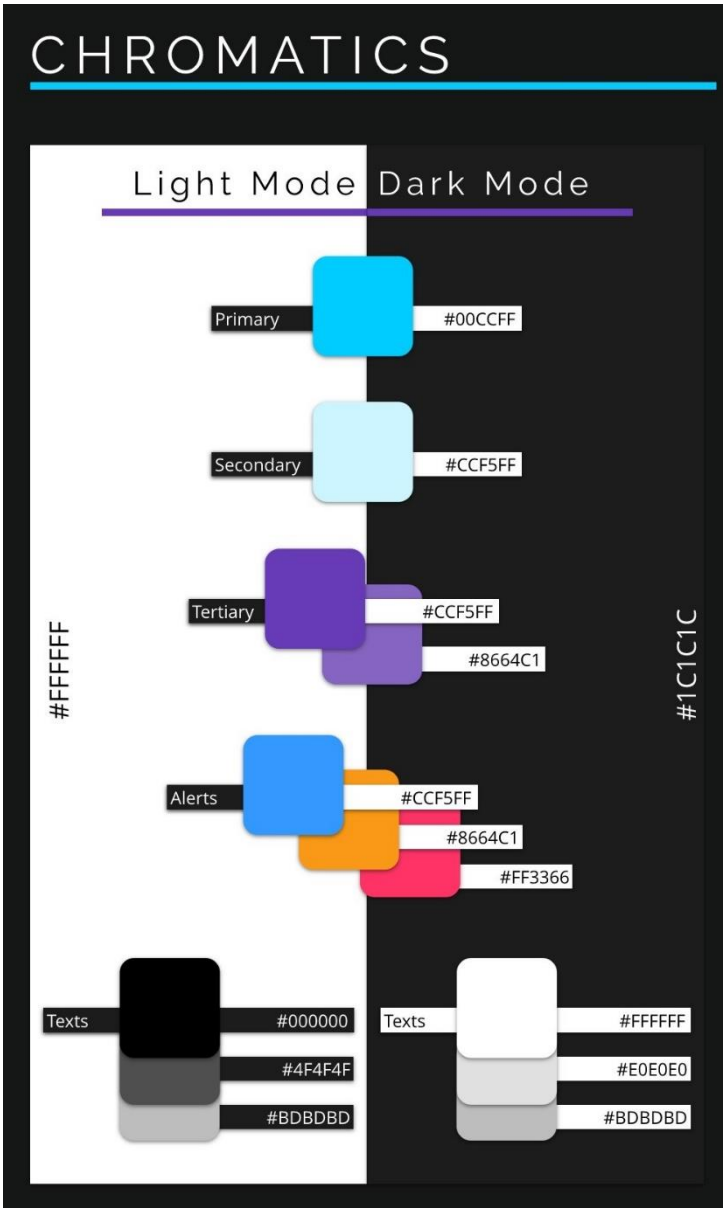


Figure 22 - Style Guide – Chromatics (Made using Invision Studio)

The term chromatics refers to the hues and saturations of colours. The theme for this project is “medical” hence the deep sky-blue primary colour, light cyan secondary colour and slate blue and medium purple tertiary colours. This colour palette is inspired by as well as representing the colours of the scrubs for the nurses in certain medical wards. They were also chosen as they all compliment both the light and dark theme, respectively.

Typography:

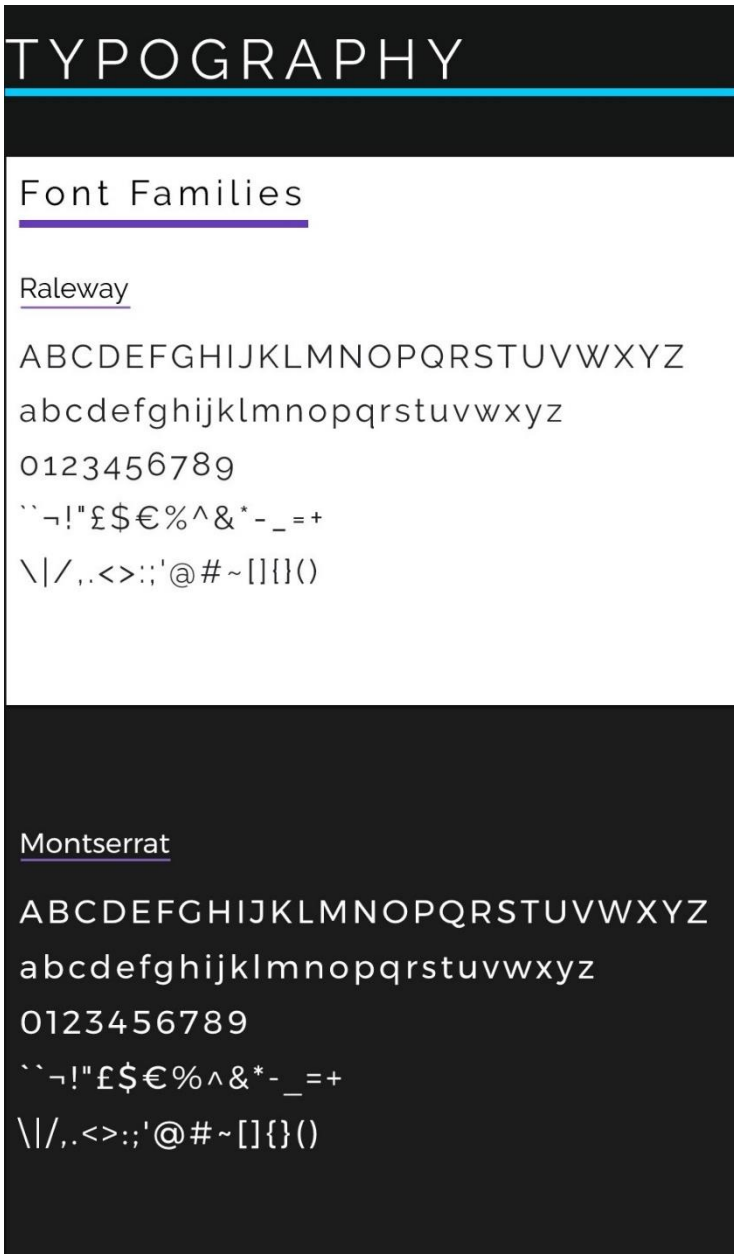


Figure 23 - Style Guide – Typography (Made using Invision Studio)

Typography refers to the technique of making letters legible and appealing. The Raleway and Montserrat font families were chosen for their contemporary style. It adds a modern look in addition to its clear readability.

Environment:

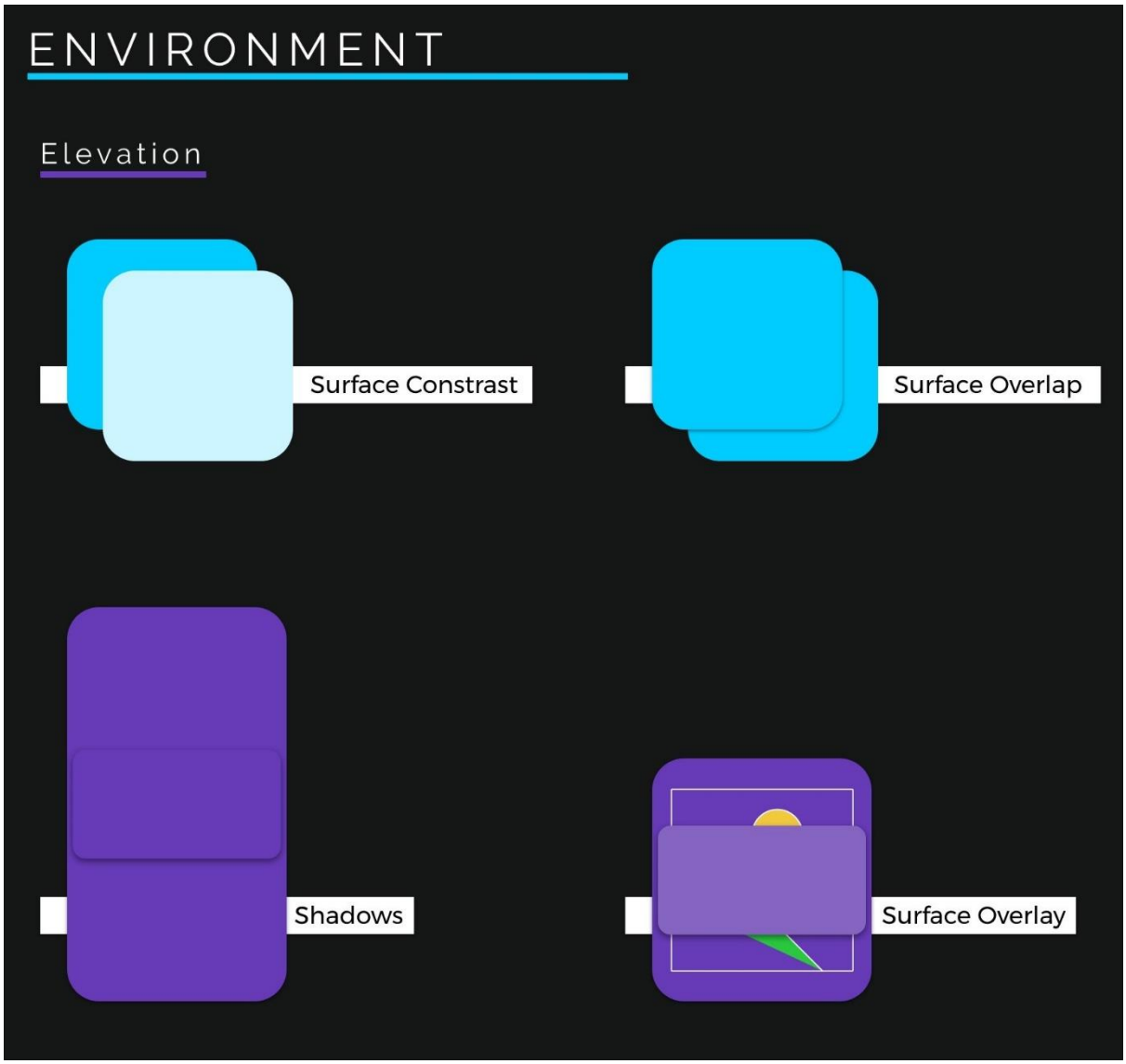


Figure 24 - Style Guide - Environment (Made using Invision Studio)

Based on Material Design (2021), the meaning of environment refers to objects being stacked or attached to one another but cannot pass through each other thus casting shadows and reflecting light. This catalogue the Google Material design system rules of elevation. It demonstrates how Material creates depth into their design. The elevation types portrayed are what is implemented in this project.

Grids and Spacing:

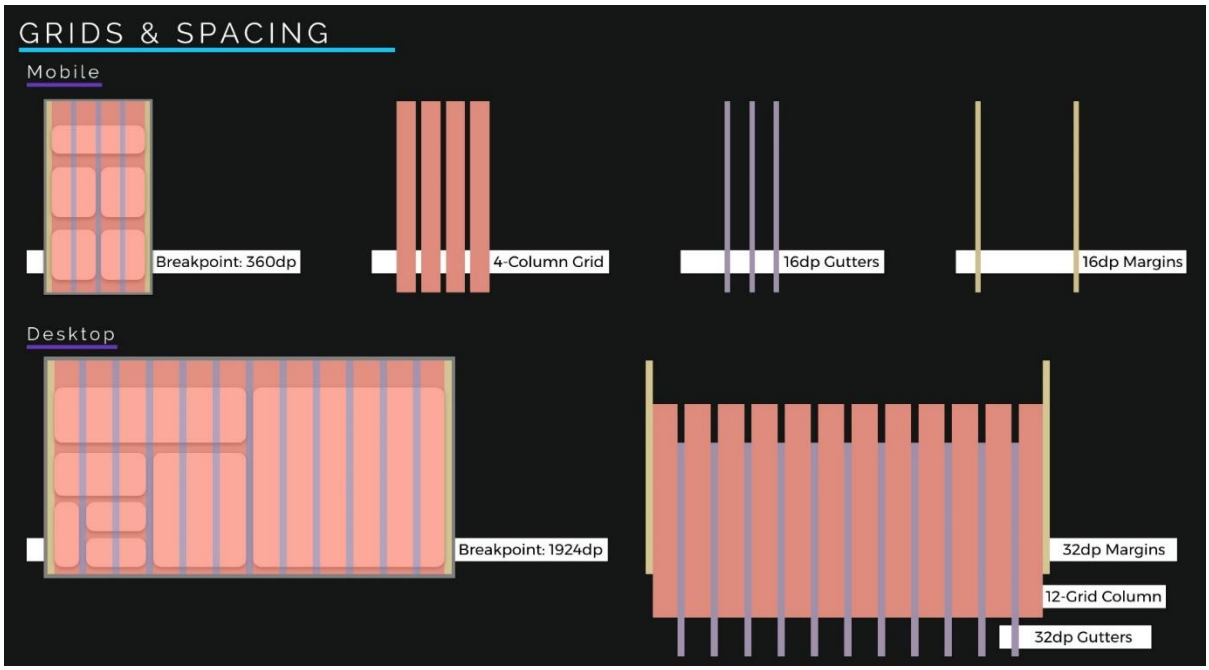


Figure 25 - Style Guide - Grids & Spacing (Made using Invision Studio)

The Material grids displayed for both smartphones and desktop act as guides for this project. It serves as an example on how contents could be spaced out.

4.3.3 Low-Fidelity (Paper) Prototype

The benefits of Low-Fidelity (Paper) prototyping is that it allows the designer to concentrate on the design and concept, it can be reiterated in real time and it is accessible to everyone.

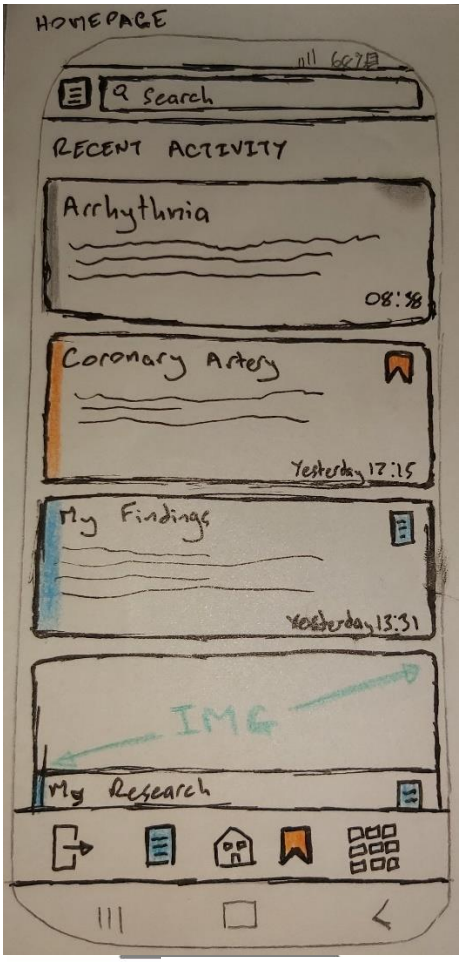


Figure 26 – Lo-Fi – Home Page

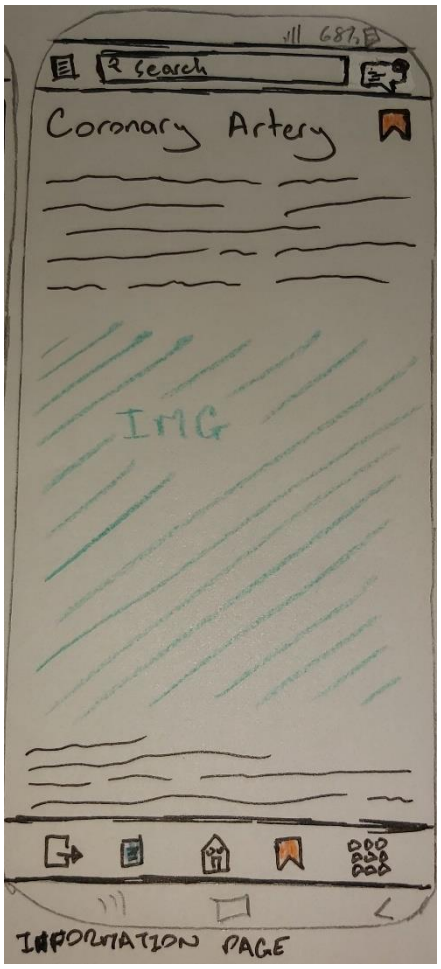


Figure 27 -Lo-Fi – Information Page

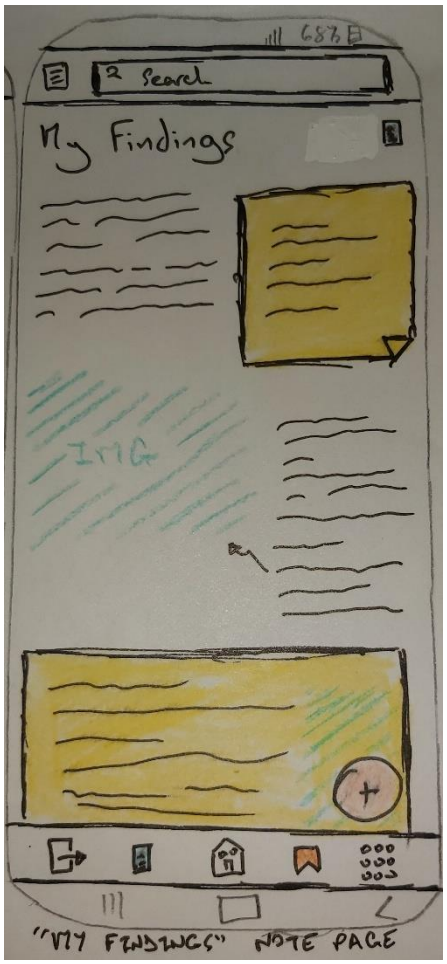


Figure 28 -Lo-Fi – Note Page

4.3.4 Wireframe

Wireframing is a valuable method to identify usability problems during the early stages of the design process. It is considered as the “skeleton” of the product which consists of two-dimensional depictions of a page’s interface. Each page displays the spacing of the elements, how content is prioritised, what functionalities and features are available, and how users will potentially interact with the product. A few examples of wireframing is seen below displaying the first iterations of the home page, information page and the notes page.

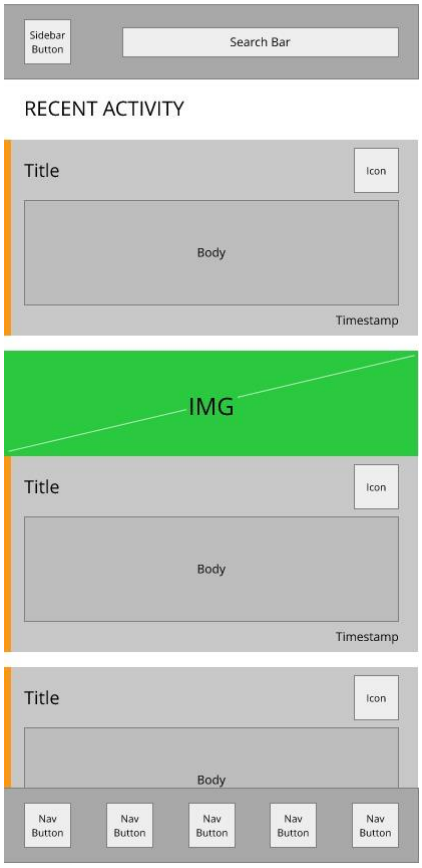


Figure 29 - Wireframe – Home Page

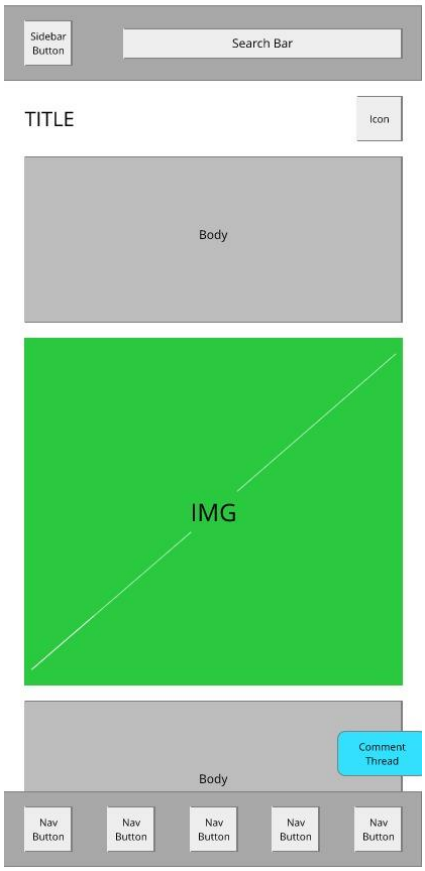


Figure 30 - Wireframe – Information Page

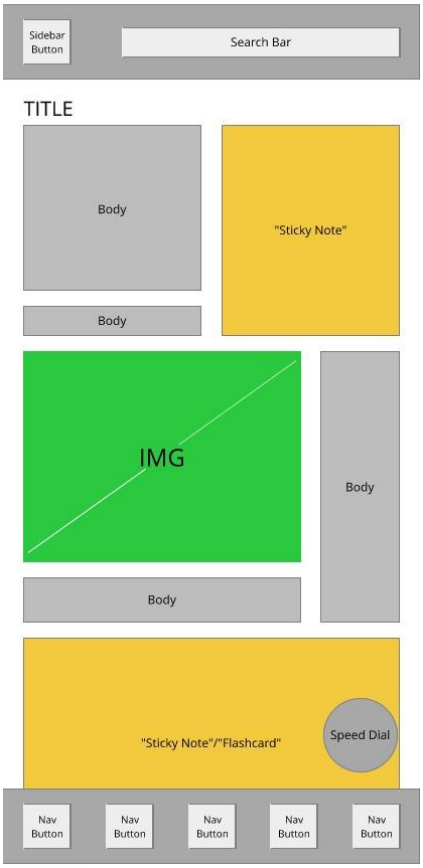


Figure 31 - Wireframe – Note Page

4.3.5 Mockups

While the wireframe is the “skeleton”, the mockup is considered as the “skin” of the product. This is when the wireframes are updated into a high-fidelity visualisation of the product, meaning, the wireframes are used to construct the aesthetics of the application. In return, once the designer has designed the mockups, this will help the development team, stakeholders, etc. to understand the final form of the product. The examples displayed below are the results of using the wireframes to create the mockups.

Figure 32 - Mockup – Home Page

Figure 33 - Mockup – Information Page

Figure 34 - Mockup – Note Page

4.3.6 High-Fidelity Prototype

The benefits of Hi-Fi prototyping are that the interface is more familiar to the users, it targets the specific components to test and in terms of the career sector, it is more presentable to the stakeholders. Hi-Fi prototypes may display the animations and aesthetics of the UI while also simulating the navigation of the users on the product all in real time. The example below depicts the process of making such prototype.

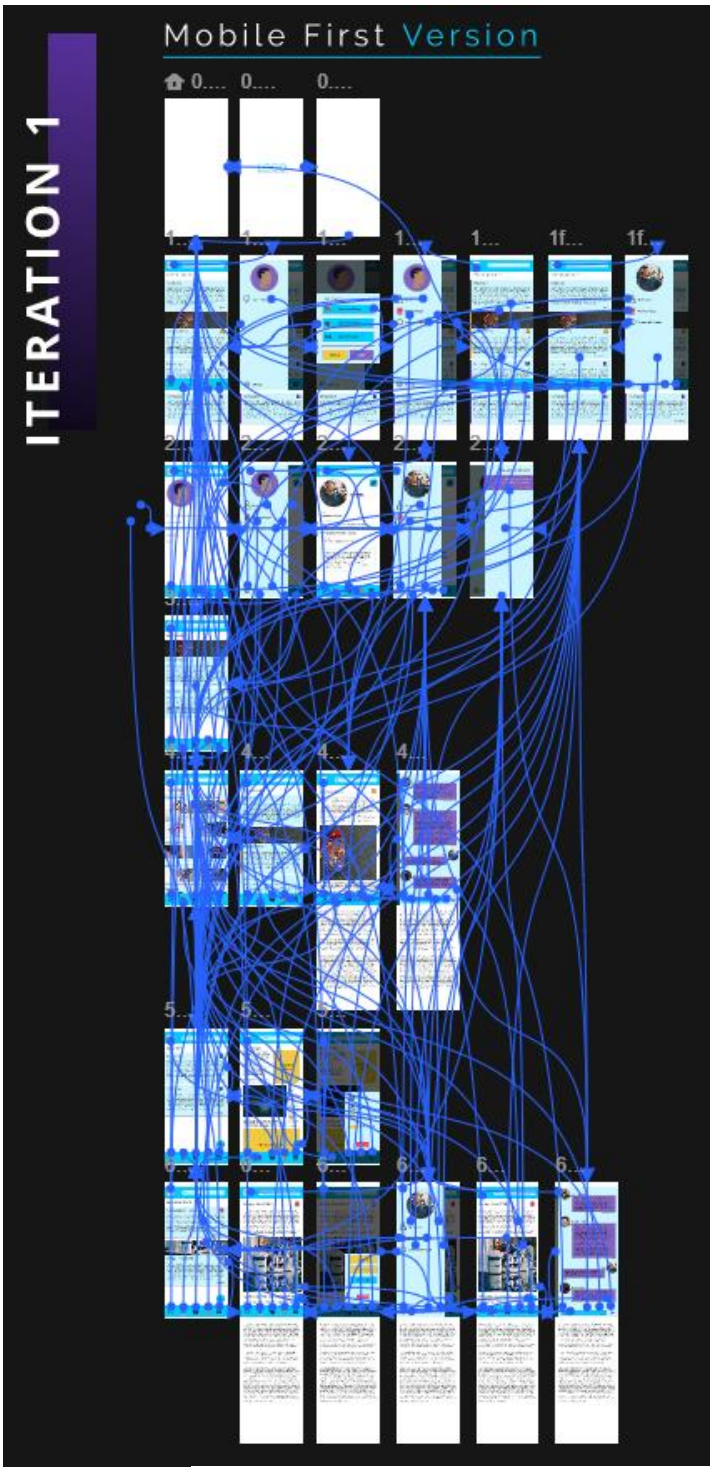


Figure 35 -Making of the Hi-Fi Prototype (Iteration 1)

(Made using Invision Studio)

Using InVision Studio, this Hi-Fi prototype was created by utilising their prototyping tool as depicted by the blue lines seen in Figure 35. The designer may select a component and transform it into an interactable section. For example, the navigation button for the bookmarks page has been connected with the bookmarks artboard using the prototyping tool.

To create sophisticated animations and/or interactions, multiple artboards must be copied and slightly changed. An example of this is for the splash page. A timer has been set on the first splash page artboard to transition to a copy of that artboard with a sample logo in it. Once the prototype is being tested, the person viewing the live prototype will witness an animation within the splash page.

4.4 Design Summary

The fundamentals of design are highly valuable to further understand and develop problem solving. Design is more than just aesthetic features. At its core, design is a system for developing meaningful and innovative solutions that creates an outcome of functional and inventive demands.

The area of this project is within the medical and education field. To fit the criteria of the medical and education space, the student must design and conceptualise a project with features of organizability, accessibility, functionality and efficiency in mind. With the chosen technologies and design patterns, it is believed that those criteria will be met at the final stage of this project. Following the application architecture, database and process design will also assist on meeting the criteria. In addition to the UI and UX, following the bespoke style guide and further iterating on concepts would possibly allow this contemporary project to become a potential product.

5 Implementation

5.1 Introduction

This chapter explains the structure of designing and developing the project to investigate the process of how this application came to be. This chapter will also discuss implementation roles, development environment, Scrum methodology (subset of Agile) and the contents of the Scrum sprints.

5.2 Implementation Roles

Full-stack Development:

An attempt was made while designing the frontend and backend structure to make the programme code as efficient as possible. For instance, within the server side, due to the Mongoose framework and its efficient methods, hooking the database to the client has been simple in comparison to linking the server to the client via Laravel. How this is done is by linking the models to their appropriate routers which are then linked to the server component. Within the server component is the call method for the database which is done by fetching the database API (i.e. MongoDB Atlas API). Multiple component files have been made and organised within their appropriate directories. This organisation method has been done to increase code efficiency. An example of this is should there be a bug on a specific functionality or feature, the developer may simply direct themselves to that component to investigate instead of attempting to find the issue in one file that has a great number of imports, functions, classes, etc.

Conversion:

The project was originally designed and developed using Atlaskit - Atlassian design system. Converting the project from its original to Google Material design was a substantial task. As mentioned in the *Technologies* section of the *Design* chapter, the decision to replace the former component library and design procedure was due to concerns of deprecation, its complexity to integrate their components to the project (as already explained within the *Technologies* section of the *Design* chapter and the fact that Material UI has the required components available to develop the application as it was designed to be.

In order to adapt the new component library, any codes and imports based on the Atlassian Design components were eradicated from the project. After the elimination, necessary installations are first imported within the project (i.e. Default and React-Material global components, Material UI Theme, Material Styles and Material Labs). After the necessary installations has been successfully imported within their appropriate files, the main conversion begins: Firstly, all of the script components within each React.js component files must be re-coded to meet the Google Material UI's mechanics. Without doing so, re-coding the template using Google Material will break the application due to not having the appropriate script components. Second of all, once the script has been converted, the classes must be converted as well. This includes the dashboard/homepage, symptom information page, notes page and the search function. Finally, once Google Material's design system is successfully displaying within the browser, the theme must be applied to style the project. This includes the chromatics, fonts, exclusive design elements and general aesthetic features.

Design:

This includes UX research, conceptualising, wireframing, paper prototyping, mock-ups, high fidelity prototyping, frontend coding, ERD iterations, UML iterations and creating a bespoke style guide. The importance of such role is to define and implement solutions that would both benefit the team and users of the medical diagnostic/reference application.

User and UX research were conducted to provide feedback on what the project would potentially become. This is done through surveys, interviews, personas, scenarios and researching within the component library's design system. Conceptualisation was done in various ways: through sketches, digital iterations and experimenting with the components by coding in a different Github branch. For this project, wireframing was drawn on A1 sheet of papers. This allowed for dozens of inspirations to be placed per page. The paper prototyping was conducted using standard white paper as well as tracing paper. This allowed for simulating the potential functionalities of the project. The mock-up, Hi-Fi prototype and style guide were illustrated in Adobe XD and InVision Studio as both has the outstanding capability that the team needed to visualise the "skin and nerves" of the project. ERDs and UML diagrams were composed in a web software called draw.io. This software had the elements needed to digitally draw out such diagrams.

Documentations and Presentations:

The practice of documentations and presentation set-ups are important abilities to become a proficient professional designer as it helps to develop good communication skills. The documents included within this project are the Requirements, Research, Design, Implementation, Testing, and the final thesis/dissertation itself. Setting up presentation slides for proposing the project is also included within this role.

5.3 Development Environment

Code Editor:

For the process of this project, Visual Studio Code was used to develop the system. The reason for this is that Visual Studio Code is lightweight and offers a lot of flexibility with its Extensions, Debugger, Integrated Github GUI, Find & Replace scripts, and CLI (Command-line interface) integrations.

The studio also offers a limitless number of extensions and themes that allows the developer to tailor it to their own development environment thus permitting the developer to drastically speed up the development time.

Git:

Git was used from the very beginning of the project. This allows the developer to extensively use version control to keep track of changes throughout the duration of the development. Git also allows collaboration between multiple developers on the project simultaneously. This software also manages any potential code conflicts in most cases automatically.

In the event that a large feature would be added (e.g. migration from Atlassian to Material design systems), decisions were made to create new branches for the large changes. Once these changes were stable, they were merged back to the master branch of the repository.

5.4 Scrum Methodology (Subset of Agile)

To understand Scrum Methodology, Agile Project Management must be defined. Agile is an array of principles and values that outlines a team's daily activities and communications. Scrum is a branch of Agile which also extends its specifications and interpretations, particularly for software development disciplines.

The goal of Scrum is to deliver new software functionalities per 2 - 4 weeks known as "Sprints". In practice, this would allow the developers to achieve higher productivity, better quality products, reduced time to market, improved stakeholder satisfaction, better team dynamics and overall develop a more organised and effective team.

By utilising Scrum for this project, complexity in building the project would be addressed by making information transparent, thus allowing others to inspect and adapt on current conditions rather than predicted conditions. This would also allow the team to address the common difficulties of a waterfall development process (i.e. chaotic results from constant requirement changes, miscalculated time management, compromises on project quality and inaccurate report about the progress). Using this methodology also allows the team to frequently inspect their work to ensure progress as well as to detect discrepancies at an early stage to give them some time to make quick adjustments.

Scrum Methodology consists of four components: The Scrum Team, Scrum Events (Ceremonies), Scrum Artefacts and Scrum Rules.

The Scrum Team:

The team contains more than two people. There are no team leaders involved to assign tasks as well as deciding on how to solve a problem/s however, the team as a collective decides on how to address the issues and how to solve them. Each member of the team is expected to deliver a concept from development to production-level.

As stated above that there are no team leaders involved, there are however, roles within the team. In an enterprise scenario, the Product Owner is the project's key stakeholder. They are the people that express what the overall assignment and vision of the product will be. They will also be responsible for managing the product backlog and accepting completed increments of work. The Scrum Master acts as the coordinator for the Product Owner, Development Team and Organisation. They are accountable to keep the team running smoothly as well as making sure that they are following the Scrum theory, practices and rules. For this project, the Scrum Master is the team's supervisor. The Development Team is a group of self-organising and cross-functional squad that are capable to deliver completed tasks per Sprint.

Scrum Events (Ceremonies):

This component contains five elements which were employed throughout the project to reflect a professional environment as much as possible:

The *Sprint*, as already mentioned, is a time span between 2 to 4 weeks to deliver new product capabilities.

The *Sprint Planning* Sprint refers to when the team holds a meeting (e.g. weekly event) which they would plan out the next move for the project as well as how to achieve it.

The *Daily Stand-up* is another meeting event which would consist of a quick communication from each team member about the progress since the previous stand-up.

The *Sprint Review* allows the Development Team to demonstrate their progress on the project during a specific Sprint. The Project Owner evaluates the work and judges whether it is acceptable or not.

The *Retrospective/Retro* directs to the final meeting within the current Sprint. It is in this meeting that will allow the team to determine what went well, what obstacles they came across and what can be improved.

Scrum Artefacts:

This contains informative documents that would outline the progress of the project. In comparison to the two mentioned below, the *Product Backlog* is considered to be the most important document. This is due to how it holds every requirement for a system, project and/or product. This backlog is also considered as a to-do list for work items, each of which produces a deliverable with business value. The *Sprint Backlog* is a specification for an exclusive list of items taken from the Product Backlog. These must be completed within each Sprints. Finally, the *Increment* is a total of every Product Backlog items that has been accomplished.

Scrum Rules:

The rules are governed by what benefits the process for the team.

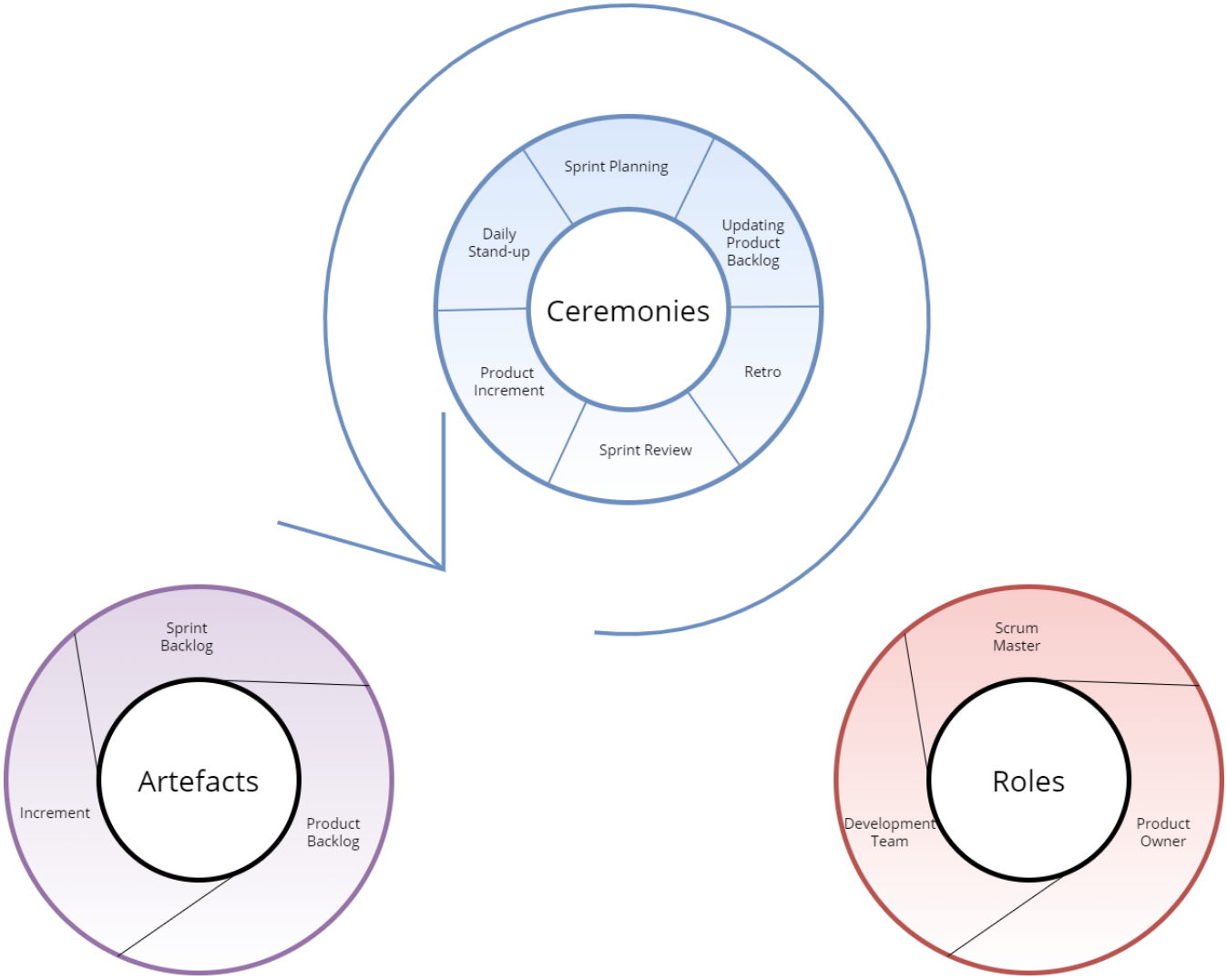


Figure 36 – Scrum Methodology

5.5 Sprints

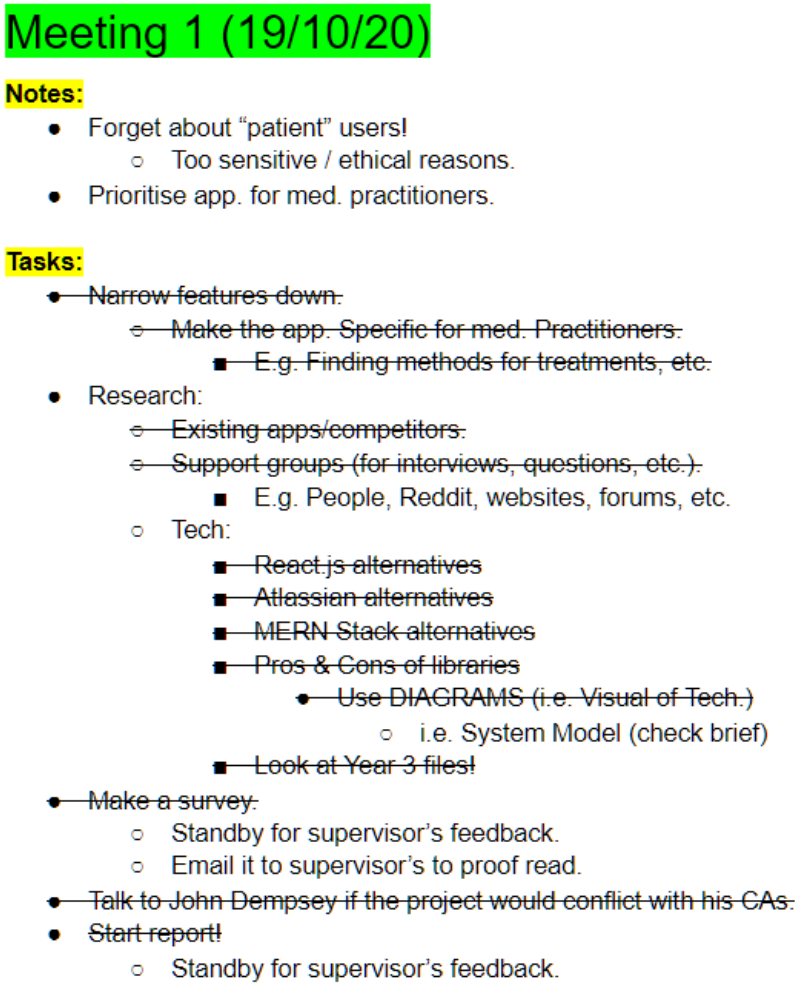
5.5.1 Sprint 1

Goals:

Sprint 1 was specifically used for the setting up phase. This is when the team decides on what, where and how to document the backlogs, progress, processes and outcomes. This is also the timeframe to plan out the structure and contents of the project as well as initiating user research.

Item 1 – Journal Entries:

The purpose of writing journal entries via Google Docs is to allow the team to document the progress as well as the next tasks for the project.



Meeting 1 (19/10/20)

Notes:

- Forget about "patient" users!
  - Too sensitive / ethical reasons.
- Prioritise app. for med. practitioners.

Tasks:

- ~~Narrow features down.~~
  - ~~Make the app. Specific for med. Practitioners.~~
    - ~~E.g. Finding methods for treatments, etc.~~
- Research:
  - ~~Existing apps/competitors.~~
  - ~~Support groups (for interviews, questions, etc.).~~
    - E.g. People, Reddit, websites, forums, etc.
  - Tech:
    - ~~React.js alternatives~~
    - ~~Atlassian alternatives~~
    - ~~MERN Stack alternatives~~
    - ~~Pros & Cons of libraries~~
      - ~~Use DIAGRAMS (i.e. Visual of Tech.)~~
        - i.e. System Model (check brief)
    - ~~Look at Year 3 files!~~
- ~~Make a survey.~~
  - Standby for supervisor's feedback.
  - Email it to supervisor's to proof read.
- ~~Talk to John Dempsey if the project would conflict with his CAs.~~
- ~~Start report!~~
  - Standby for supervisor's feedback.

Figure 37 – Week 1 of Journal Entry

Item 2 – Trello Board:

A Trello board has been constructed to achieve optimum workflow management.

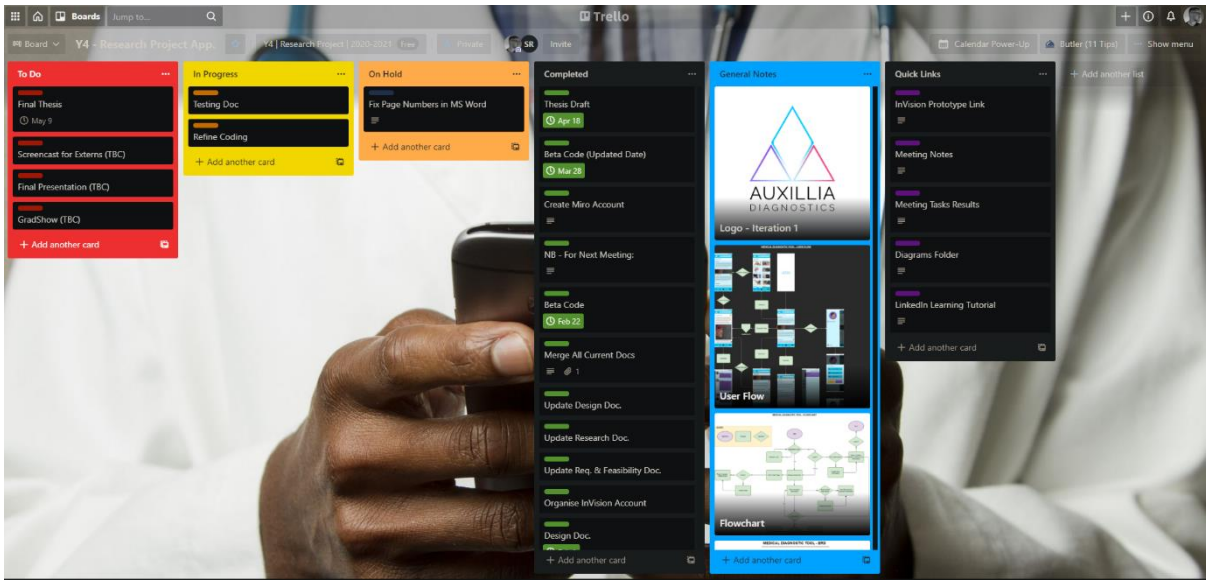


Figure 38 – Trello Board

Item 3 – Initialising GitHub Repository:

A repository has been made to contain the project as well creating various branches to develop experimental features.

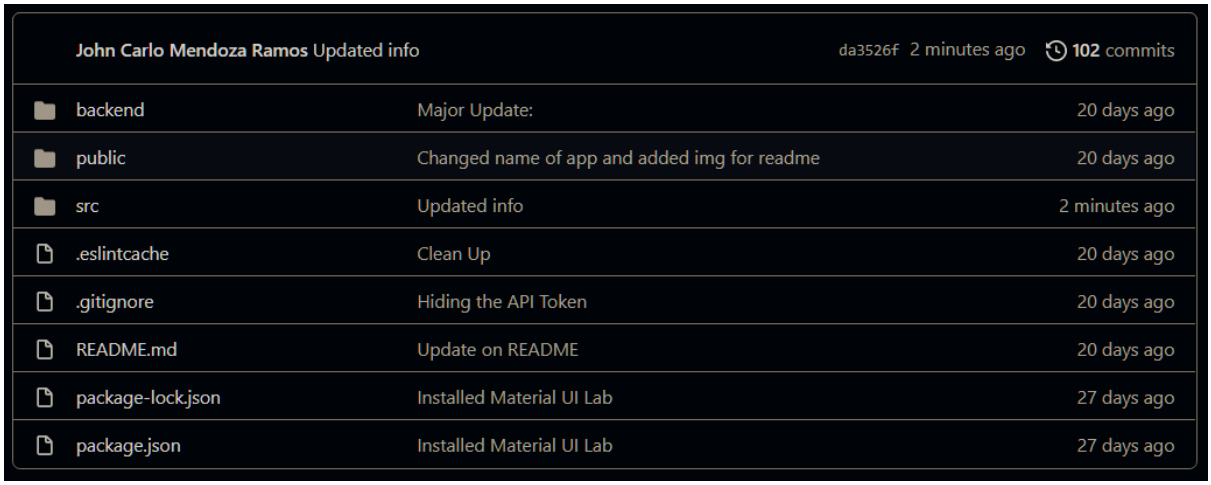


Figure 39 – GitHub Repository

Item 4 – User Research:

Data was obtained via quantitative and qualitative methods. A survey and interviews were conducted to collect various information from the target audiences (i.e. medical staff and students) and personas and scenarios were also created to achieve ideas and problem statements.

Item 5 – Entity Relationship Diagram (ERD):

An Entity Relational Diagram (ERD) was designed to display the relationships of entity sets that are stored in a database.

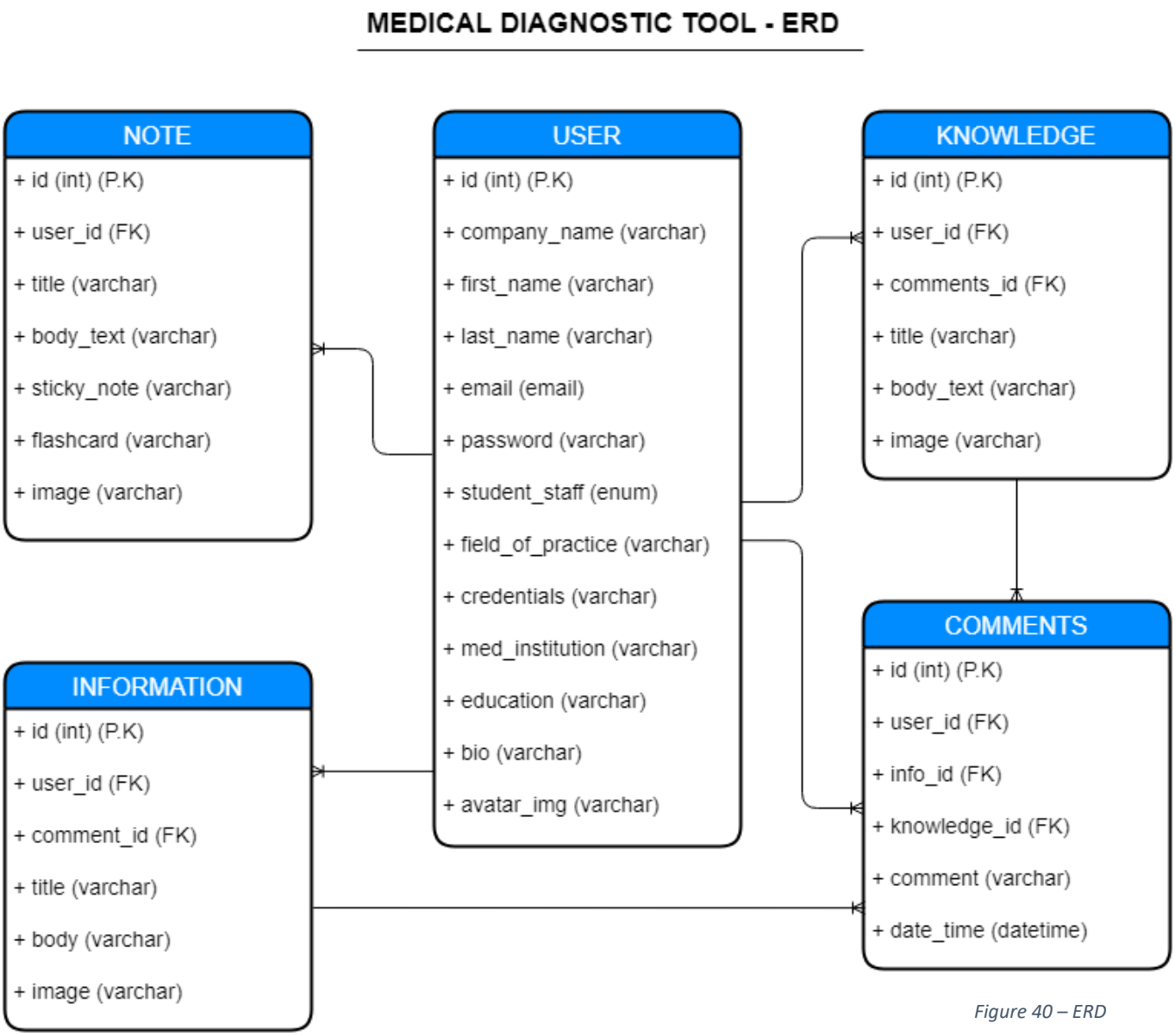


Figure 40 – ERD

Item 6 – Initialising Server-side:

Using the ERD, models, routers and the server were coded into the project within the backend. This allows the database to be migrated and seeded with data implemented within the ERD.

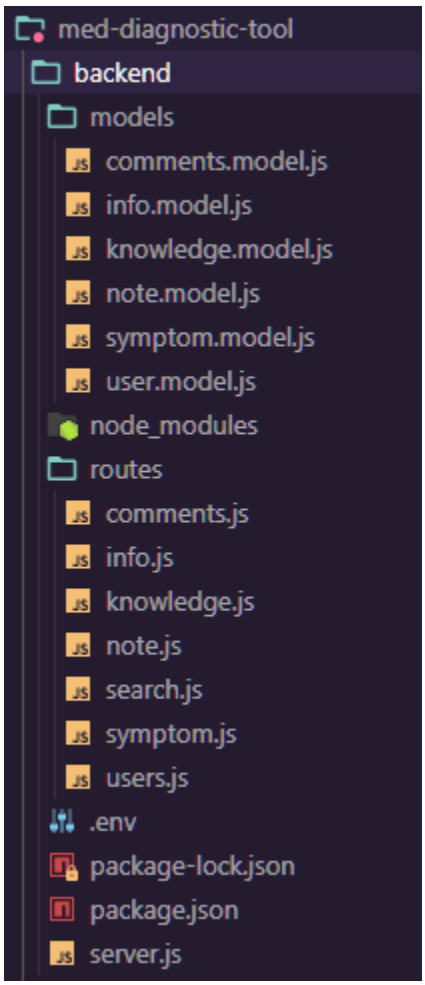


Figure 41 – Models & Routes components as well as Server component

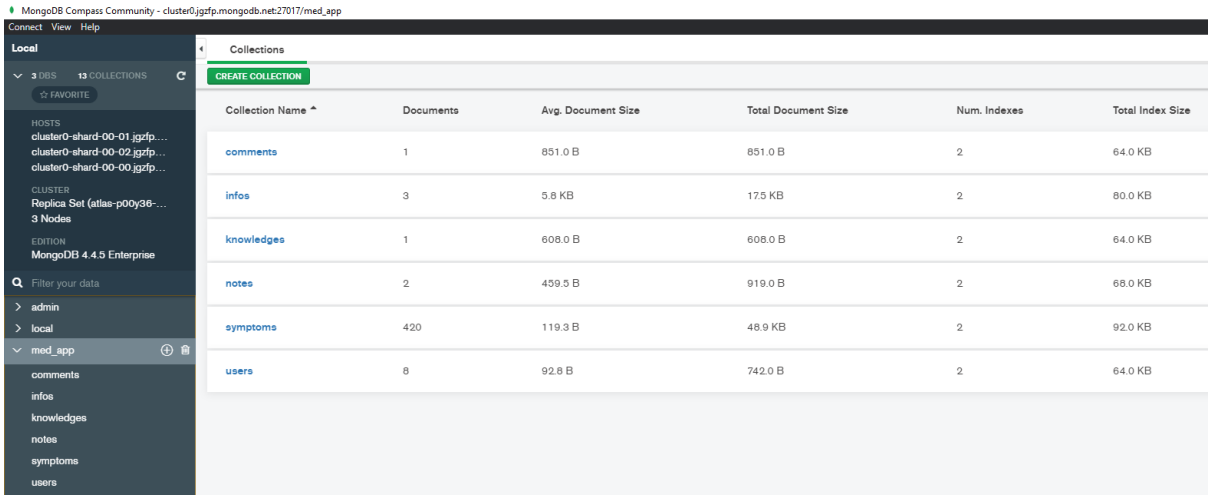


Figure 42 – MongoDB Atlas (via MongoDB Compass) Database

Item 7 – Fetching API:

There are three API requests within this project: The first is an API request for MongoDB Atlas within the server component, the second is a manually made JSON API for testing and debugging, and the last is the *System Checker* API for deployment (see Figure 43).

```
const uri = process.env.ATLAS_URI;
mongoose.connect(uri, { useNewUrlParser: true, useCreateIndex: true });
const connection = mongoose.connection;
connection.once('open', () => {
    console.log("MongoDB database connection established successfully");
});
```

Figure 43 – MongoDB Atlas API fetch request (Note: "ATLAS_URI" is the variable name which holds the API key)

```
componentDidMount() {
  axios
    .get("http://localhost:5000/Symptom/")
    .then((response) => {
      this.setState({ symptoms: response.data });
    })
    .catch((error) => {
      console.log(error);
    });
}
```

Figure 44 – Tester API fetch request

```
componentDidMount() {
    axios.get(BASE_URL + "/symptoms?language=en-gb&format=json", {
        headers: {
            "x-rapidapi-key": TOKEN
        }
    })
    .then(response => {
        console.log('response!', response)
        this.setState({ symptoms: response.data })
    })
    .catch((error) => {
        console.log(error);
    })
};
```

Figure 45 – Symptom Checker API fetch request

Item 8 – Initialising React's App (Main) Component:

A single file that acts as the main component for the project was created to jump start the frontend development of the project.

src > App.js > ...

```
function App() {
  return (
    <Router>
      <MuiThemeProvider theme={theme}>
        <div className="App">
          {/* Navigation Components */}
          <Nav />
          <Navigation>

            <Switch>
              {/* Symptoms Information Components */}
              <Route path="/" exact component={Homepage} />
              <Route path="/create" component={CreateInfo} />
              <Route path="/view/:id" component={ViewInfo} />
              <Route path="/edit/:id" component={EditInfo} />
              <Route path="/user" component={CreateUser} />

              {/* Notes Components */}
              <Route path="/note" component={NotesPage} />
              <Route path="/create/note" component={CreateNote} />
              <Route path="/view/note/:id" component={ViewNote} />

              {/* Bookmark Component */}
              <Route path="/bookmarks" component={BookmarksPage} />

              {/* Testing API Component */}
              <Route path="/symptom" component={SymptomsPage} />
            </Switch>
          </Navigation>
        </div>
      </MuiThemeProvider>
    </Router>
  );
}
```

Figure 46 – View of App (Main) Component

5.5.2 Sprint 2

Goals:

As the goal for Sprint 1 has been met, Sprint 2 is set to achieve the conceptual frontend design of the project. The designs were then coded and updated at further dates.

Item 1 – Wireframes:

Wireframes were created to visualise the potential interface design of the application.

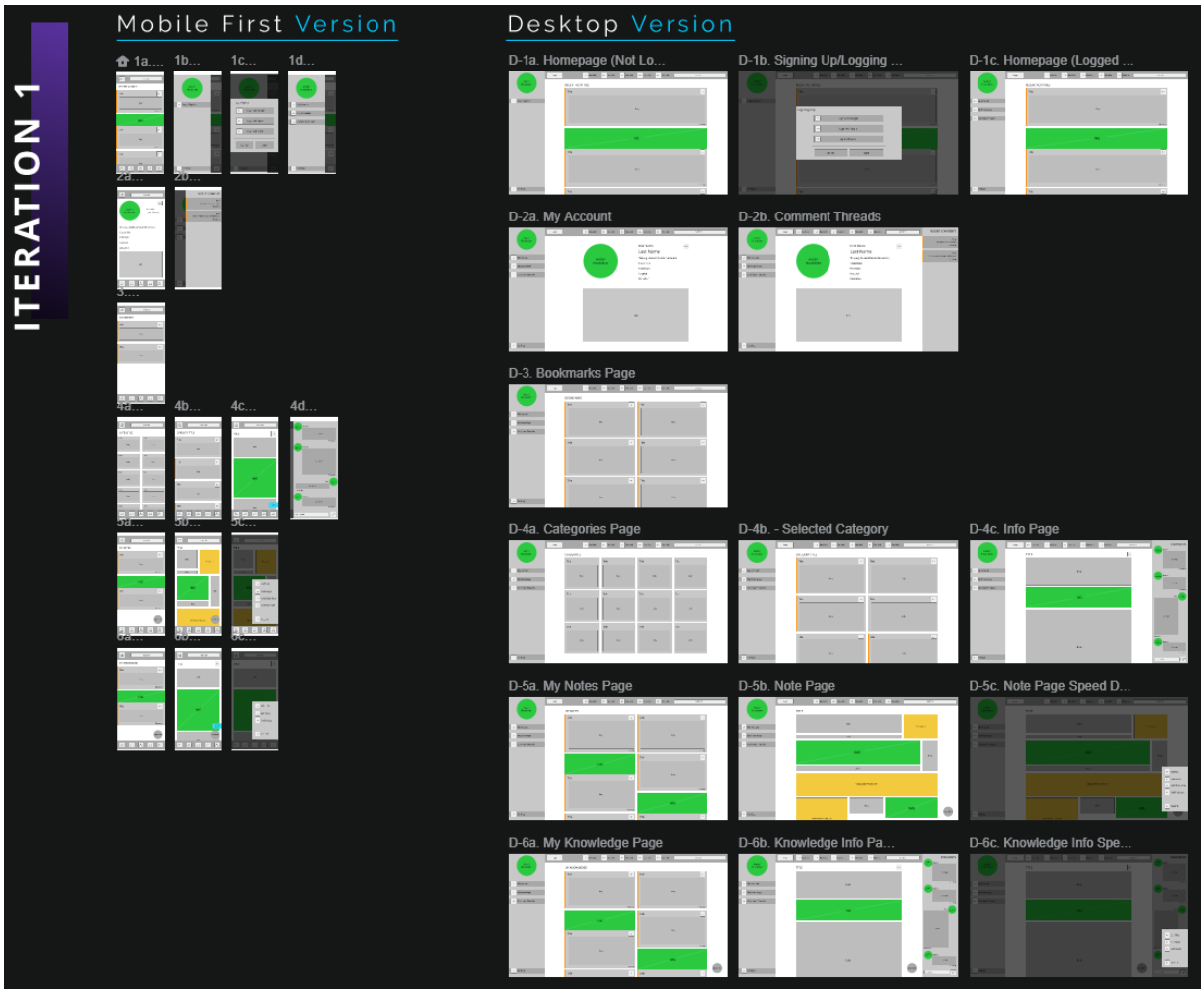


Figure 47 – 1st Iteration of Wireframe

Item 2 – Paper Prototypes:

These were created to conceptualise the potential functionalities as well as the layout of the application.

Item 3 – Flowcharts:

A flowchart was made to assess options that the user may have within the application.

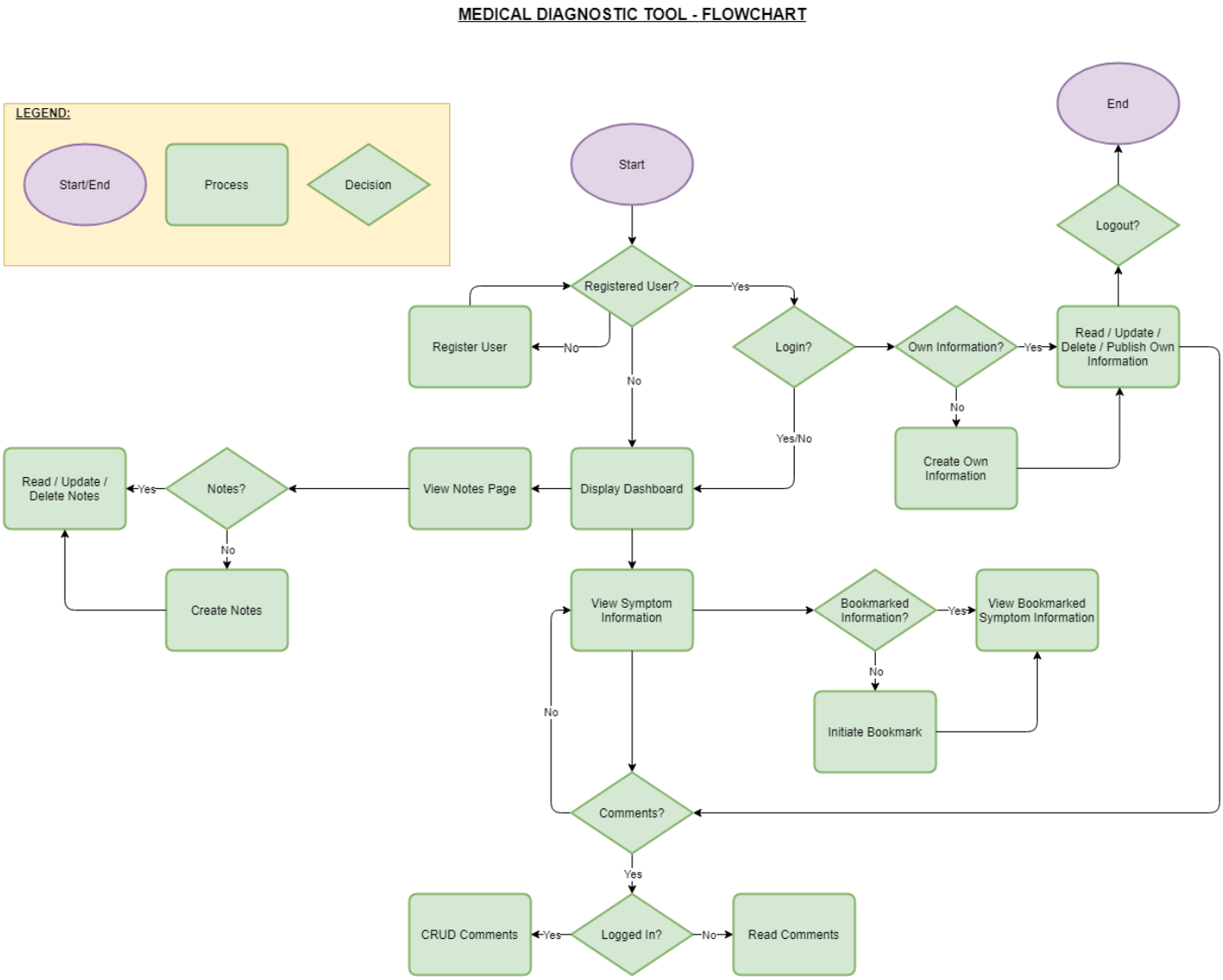


Figure 48 – Flowchart

5.5.3 Sprint 3

Goals:

Sprint 3 allows the designer to further develop the design iterations. Obtaining U.X inspirations and conducting UX research (see *Research – Literature Review of Interaction Design* chapter) were also conducted within this Sprint.

Item 1 – High Fidelity (Hi-Fi) Prototypes:

The iterations for the Hi-Fi prototypes were made using InVision Studio. By designing the mock-ups and portraying an accurate prototype (in comparison to the paper prototype), it permits the designer to envision the project with attention to detail in mind.

Item 2 – UX Research:

Based on the chosen component library (i.e. Google Material Design System), research has been conducted to access guidance on how to design an application with user satisfaction as the goal. UX research on interaction design has also been documented within this Sprint to gain further knowledge to use within the design of the application.

Item 3 – Custom Style Guide:

A bespoke style guide has been designed to further support the UX research. This style guide was also used to aid with development of the frontend for the project. Note that the style guide seen below was made for the implementation of Google Material Design System.

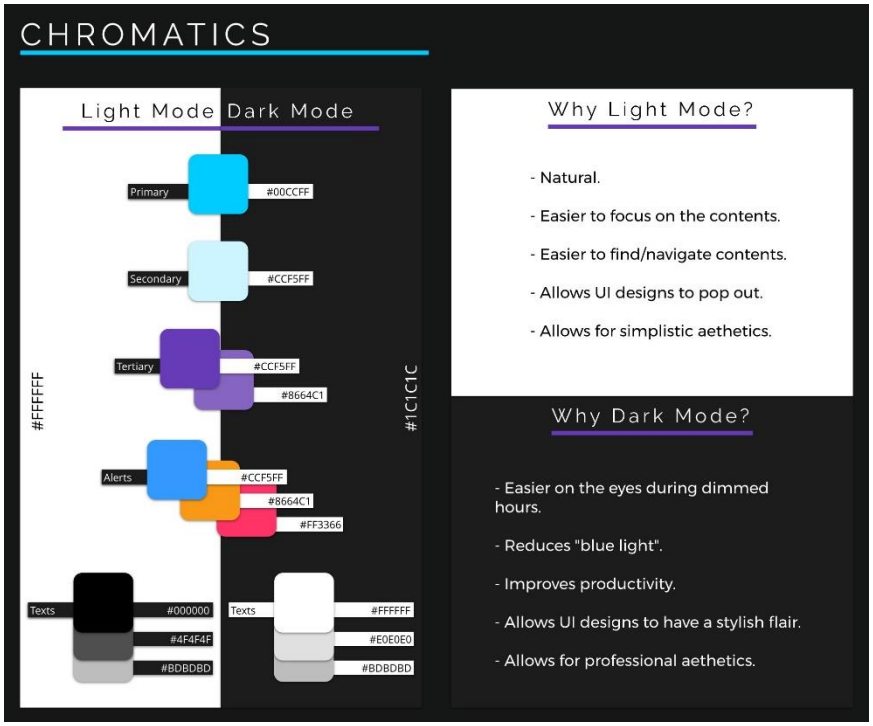


Figure 49 – Style Guide for Chromatics)

5.5.4 Sprint 4

Goals:

This timeframe was used to dedicate on documentations as well as updating the UML diagrams.

Item 1 – UML Diagrams:

Aside from updating the information within the Flowcharts, a User Flow diagram was created to depict an accurate variation of the Flowchart.

5.5.5 Sprint 5

Goals:

The goal for Sprint 5 was to convert the project from Atlaskit – Atlassian Design to Google Material. Considering that this was a large task, this was the main focus during this timeframe.

Item 1 – Conversion:

Converting from one component library to another is a substantial task. This is due to the difference of coding structures, design elements and philosophy as well as documentations. In consideration to its learning curve, Sprint 5 was dedicated for this item.

5.5.6 Sprint 6

Goals:

The goal for this Sprint is to further improve the functionalities of the project as well as adding new features.

Item 1 – Search Functionality:

Using Material UI's "autocomplete" documentations, the search functionality of the application was created during this timeframe.

5.5.7 Sprint 7

5.5.8

Goals:

The final Sprint consists of the final implementation of a new feature as well as submitting the Alpha build of the project. This refers to all functionalities and styling planned must be implemented within the codes and documentations.

Item 1 – Personal Note Taking:

This final function implemented within the project is the note taking functionality. This will allow users to take personal notes within the application and will be saved within their dashboard as well as the notes pages. The user may then create a new one, view their notes, update their notes or delete them (i.e. CRUD functionality).

5.6 Implementation Conclusion

With all the features and styling now implemented, the project was taken shape into what it was designed to be. To give the project identification, the application has been officially named as *Auxillia Diagnostics*. This application is designed and developed to be a medical diagnostic and research tool that allows medical staff and students to quickly search up information about symptoms, their causes and their appropriate diagnosis. *Auxillia Diagnostics* has been designed to achieve the goal of creating an environment to obtain optimum workflow.

6 Testing

6.1 Introduction

Two categories of testing have been conducted for this project: functional testing and user testing. Testing refers to the process of evaluating a product or its elements with the intent to figure out whether the product meets the satisfactory level or not. It is done to ensure the reliability of the product as well as find any bugs and issues within the software. Testing is crucial before the final release of the product so it may provide the users the best user experience as much as possible from day one of publication.

6.2 Functional Testing

This type of software testing consists of validating the system against the functional requirements of the product. This allows for testing each individual function by supplying appropriate inputs and observing the outputs against the functional requirements. The two functional testing done for this project are for the navigation and the CRUD functionalities of the medical tool.

6.2.1 Navigation

Test No.	Description of Test Case	Input	Expected Output	Actual Output
1	Navigating through the app. using the right panel.	Clicking/tapping on each nav link.	Input must direct to appropriate pages.	Input has directed to appropriate pages.
2	Navigation of dashboard.	Clicking/tapping on card button to display selected content.	Input must direct to appropriate content.	Input has directed to appropriate content page.
3	Navigation of search bar.	Typing/texting certain characters within search bar.	Typing/texting certain characters must initiate predictive text function as well as providing a list of options in real time.	Typing/texting certain characters initiates predictive text function as well as providing a list of options in real time.
4	Navigating through the categories page.	Clicking/tapping on certain symptom category.	Must display appropriate symptom information based on category.	Displays appropriate symptom information based on category.

6.2.2 CRUD

Test No.	Description of Test Case	Input	Expected Output	Actual Output
1	CRUD functionalities for bookmarking.	Create, display, edit and delete bookmarked symptom information page.	Must be able to successfully perform functions.	Adding a bookmark is currently unsuccessful. Bug fixing is in progress as of writing.
2	CRUD functionalities for note taking.	Create, display, edit and delete personal notes.	Must be able to successfully perform functions.	Able to successfully perform all functions.

6.3 User Testing

This type of usability testing provides the designer with an understanding of how users interact and behave with the product. The data gained from this is then be used to improve the product's features, save time and increase user satisfaction. Included within this testing are research and information about the aim of the application, participant tasks, the participants, the reason for ease-of-use testing, recommended test environment, test preparation and finally the results from the user testing.

6.3.1 Aim of the Application

There are many medical diagnostics/referencing applications out there (e.g. *Symptomia*, *Critical Care ACLS*, etc.). However, some of these applications are exclusive to one operating system, not user friendly and does not cater to all kinds of medical staff and/or students. Additionally, there are many websites that allows users to research medical information (e.g. *Mayo Clinic* website) with ease which in turn, may cause this application to be redundant. However, the selling point of this application is the ease of access from a responsive application that can be both used within the user's desktop and mobile devices while still retaining the contents that they are currently viewing (e.g. how a user can play music in *Spotify* in one device which then updates in real-time to their other device should they be using both of them simultaneously).

Based on the user research data, the experiences medical staff and students face is that researching information about symptoms may take longer than expected. Within the medical and educational industry, this may not be good enough due to the fast-paced nature of their field. The aim of having a reliable system that provides medical personnel the fastest way to get the information that they require must be met to solve the competitors' interaction design problems.

To summarise, the idea of this project is important for medical staff and students alike. For staff members, this application will provide them with a faster way to search up a query for a symptom that may not be sure of or to do further research within their free time. For students, this application will provide them access to research materials that may require as well as the added functionality to create quick personal notes. What both kinds of users have in common is, as said before, they belong in a fast-paced environment in which they cannot afford to waste too much time.

6.3.2 Participant Tasks

The three tasks that the designer will request the tester to do would aid the design and development of the application. Once the testers are doing with the tasks, they are observed on how they would do them. The results gained from such examinations will then benefit the growth of the medical diagnostic/reference tool, hence how the tasks will help with the design and development. As mentioned in the *Aim of the Application* above, considering that this application is targeted for medical staff and students, the targeted testers will be the same.

Task 1:

The participant has been requested to demonstrate how they look up information that they require to research (i.e. information about symptoms and how to appropriately diagnose them). This would inspire the further design and development of the search functionality of the application. It would also determine the format of the app (i.e. how to structure of the information pages, the dashboard, etc.).

Task 2:

The participant has been requested to demonstrate how they record the obtained information that they have researched (e.g. bookmarking the webpage, report-style writing, quick note-taking, etc.). This would influence the updated design and development of the format for the bookmarking system of the application.

Task 3:

The participant has been requested to demonstrate on how they would do *Task 1* and *Task 2* on another device (e.g. should the user use a laptop to do the above, they will require to repeat within their mobile device vice versa. The designer/overseer will provide the participant with a laptop and/or mobile should they need it to complete the tasks.) This will allow the designer to explore various possibilities on the responsive feature of the application.

6.3.3 The Participants

As specified in both *Aim of the Application* and *Participant Tasks*, the targets are within any kind of medical staff and any field of medical students. The reason why these categories are compulsory is due to how the application is designed and developed for the medical and education industry. In essence, this application is made for gaining further knowledge via researching. These specific group of participants are required to meet the medical and educational needs as well as using the data to tailor the application's workflow to become as efficient as possible.

6.3.4 Performing Ease-of-Use Testing

It is debatable that successfully performing ease of use within the application would have a psychological effect on the user's muscle memory of the application, thus affecting the ease of learning as well. This is the reason why the designer has decided that testing should lean towards ease-of-use instead of specifically ease of learning. Ease of use testing would not just benefit with the user's navigation within the application at a subconscious level, it will also be advantageous into designing an outstanding workflow which would then induce a potential satisfying user experience (UX).

Having a medical diagnostic/reference application that has all the features that a medical staff and student may potentially need as well as designing it to be as simple and pleasing to use as possible would benefit the users' satisfactory level. It would also encourage them to keep using the application as well as attract more customers to utilise it. The three tasks mentioned in *Participant Tasks* depicts such ease-of-use testing and how the data gained from each test could be used to improve the application's design.

6.3.5 Recommended Test Environment

A recommended test environment would be anywhere that is comfortable and natural. This refers to an area that can be used to easily have a normal conversation with someone. It is believed that a lecture theatre, a conference room, an office or any other space that would isolate the designer and tester together is not recommended. This is due to how it could lead the tester to become tense and unnatural with their answers thus potentially gaining inaccurate results.

Design principles such as chromatics, visual hierarchy, visual grammar, clarity and invisible design also plays the part of choosing the right environment to make the user as comfortable as possible during testing or even interviewing. By utilising user manipulation to the designer's advantage, they could gain valuable and reliable data from the testers by manipulating them to become relaxed.

An example of such environment may be within a café. This is due to how café environments are the norm for social interactivities and physical communication in a relaxed manner. This can be used to have a carefree conversation with the participant as well as offering them a beverage to increase their comfort level. As the relaxed atmosphere is steady, that consistency will carry over through the testing phase. Having the test in a public area would release the tension between the designer and tester as the environment itself is informal and casual.

6.3.6 Test Preparation

This consist of the introduction, consent form, pre-test questionnaire and post-test questionnaire (see *Appendix F – Test Preparation for User Testing*). The introduction contains what the interviewer will say to the participant. This consists information of the test, such as what the prototype is and what the participant will be doing with the prototype. This will also give the opportunity to make the participant feel as comfortable as possible during the test.

It is agreed that some prepared documents are vague (particularly within the *Post-test Questionnaire*). This is due to how the interview within the usability test will be a semi-structured interview. The reason for such decision is to make the interview less scripted and more natural, thus simulating a nonchalant conversation which would add to the relaxed atmosphere.

6.3.7 Results from User Testing

To summarise, the user testing participant has been pleased with the overall aesthetic design of the prototype. However, in acknowledgement of the feedback obtain from the participant, the list below has been created to display what requires improvements:

- The predictive text feature of the search bar only activates once a good number of letters has been typed/texted in (see Figure 50).
- The list of options after the predictive text has been activated is successful.
- Dashboard contents on desktop version is too wide and big. Potentially splitting the cards into two columns may fix this.
- Text formatting on symptom information pages is too small and not paragraphed properly thus making it slightly mundane to read.
- Notes page layouts is too basic (i.e. the same layout as symptom information pages). Slightly redesigning the layout may differentiate it from the symptom information page to give it its own unique characteristics.
- Users do not need to login/logout manually every time they re-visit the application. One sign-up and automatic login or removing the user authentication altogether should meet the theme of ease of accessibility and ease of use.
- No pagination for all contents (i.e. dashboard, categories page, notes page, and bookmarks page).
- No indication in dashboard whether the card contents are bookmarked contents, notes contents or recently visited contents.
- Bookmarking feature is buggy.

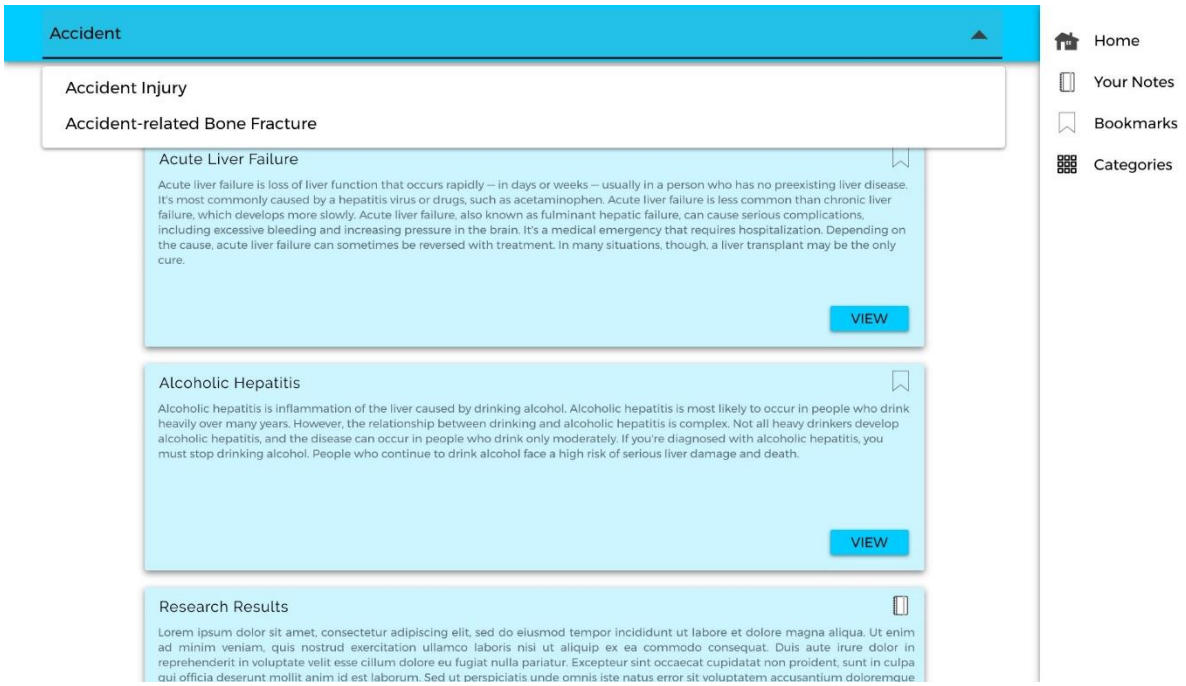


Figure 50 – Predictive text feature of search bar only activates once the user inputs a certain amount of letters

6.4 Testing Conclusion

This chapter has discussed on the importance of both functional and user testing. The functional testing has aided the developer to further improve on the application's mechanics. It also allows for organising the contents (such as what is working and what is not) which permits the developer to develop the application within an efficient workflow. On the other hand, the user testing has aided in deciding what could be improved on based on the user experience (UX) of the participant. It has also helped with the practice of testing a product, whether it is a prototype, alpha stage, beta stage of early production.

The unfortunate pandemic outbreak has made it difficult to conduct a formal usability test with various kinds of users. While there is the ability to test remotely/digitally (which has been done for this application), it does not feel that it is right to perform such method in comparison to the official way for conventional practices reasons. Additionally, benefits for conducting a face-to-face usability test would also build up the communication and professionalism skills.

7 Reflection & Conclusion

The main goal of this project was to design and develop a medical diagnostic and reference tool. The intention of this application was to serve as a researching tool that any kind of medical staff and any field of medical students may use to look up information quickly and efficiently about symptoms, causes and diagnosis.

In early stages of the project, a comprehensive analysis for requirements was performed to determine the features, functionality and design of the application. Those requirements were founded based on quantitative and qualitative data. A detailed research documentation on interaction and user-centred design were carried out to support the design and development of medical and educational application system. All the information gained from that UX research has been used in practice to design this medical and educational application to achieve an optimum of user satisfactory as well as efficient workflow.

This project has been considered multi-disciplinary due to the addition of UI design, UX design and research, data collection, analytics and web development. Admittedly, more work has been done towards the UI/UX design and research in comparison to the development. This is due to the skill differentiation and focus of the student, meaning, the student leans more towards the designing discipline in comparison the developing discipline. Considering this as well as how this project has been accomplished by one student, the project will require an extensive number of improvements both in terms of designing and particularly developing.

Completing a large software development project has improve the skill of time management, programming, designing, researching and problem solving. It had also allowed the expression of one's current design abilities and challenge the coding capability which would then further improve on those key skills. This has been more prominent for this year in comparison to last year's project which was team-based.

Hard, consistent and dedicated work has been carried out into making what the application is today. However, the usability testing has identified some enhancements and changes that would be required before the application could be released. For instance, the UI may be updated to be more pleasing for the user by removing the fixed side panel in desktop mode. This way, the user may focus specifically on what they are reading. Additionally, a functionality to filter through information based on the user's field (e.g. the oncology ward) may also be added.

In conclusion, a considerable amount of knowledge has been acquired during this project. The knowledge gained is to be used to improve current skills and learn new ones to face the upcoming challenges that awaits.

7.1 Final Application

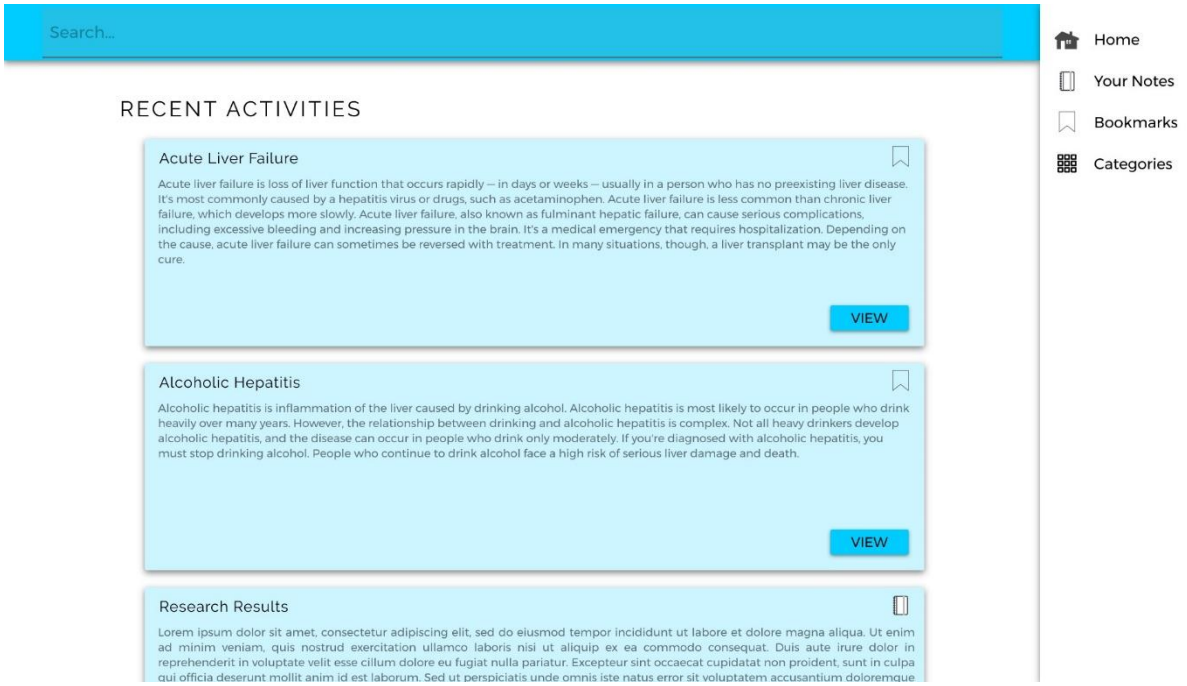


Figure 51 – Final Design of Homepage (Desktop version)

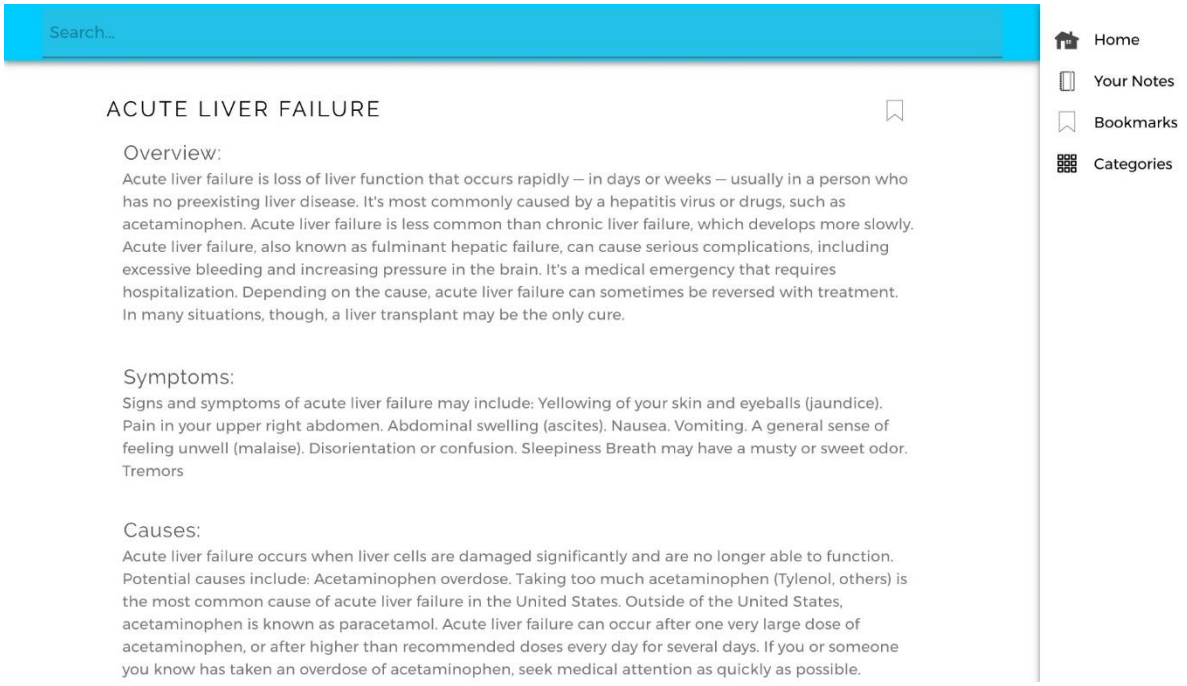


Figure 52 – Final Design of Symptom Information Page (Desktop version)

Figure 53 – Final Design of Homepage Page (Mobile version)

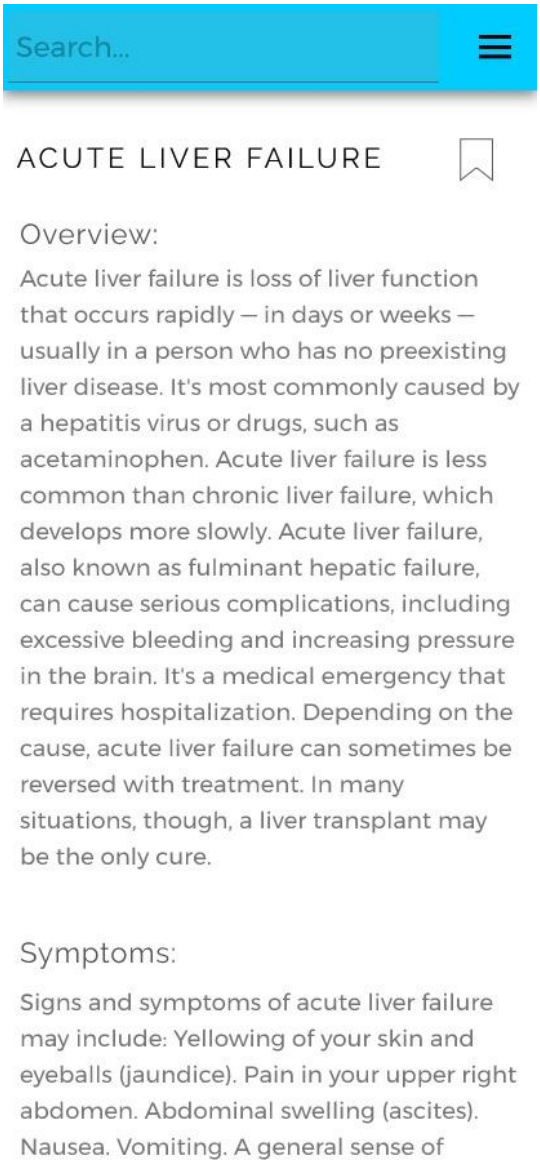


Figure 54 – Final Design of Symptom Information Page (Mobile version)

8 References

4 Reasons Why Product Testing Is Important for Manufacturers. (2018). Applerubber.com. https://www.applerubber.com/blog/4-reasons-why-product-testing-is-important-for-manufacturers/#:~:text=To%20ensure%20quality%20and%20reliability,are%20safe%20for%20customer%20use.

Atlaskit by Atlassian. (2021). Atlassian.com. https://atlaskit.atlassian.com/packages%20Atlassian%20Design%20System.%20(2020).%20Atlassian%20Design%20System.%20https://atlassian.design/

Build software better, together. (2021). GitHub. https://github.com/

Cesare Ferrari. (2019, November 27). *Working with Axios in React*. DEV Community; DEV Community. https://dev.to/cesareferrari/working-with-axios-in-react-540c

Costa Coffee Ireland. (2013). Costaireland.ie. https://www.costaireland.ie/

Deprecation notice - Basic authentication with passwords and cookie-based authentication. (2019). Atlassian.com. https://developer.atlassian.com/cloud/jira/platform/deprecation-notice-basic-auth-and-cookie-based-auth/

Design Patterns - MVC Pattern - Tutorialspoint. (2021). Tutorialspoint.com. https://www.tutorialspoint.com/design_pattern/mvc_pattern.htm

diagrams.net - free flowchart maker and diagrams online. (2021). *Flowchart Maker & Online Diagram Software*. Diagrams.net. https://app.diagrams.net/

Digital.ai resource library. (2021). Digitial.ai. https://digital.ai/resources/library

Divyani Tyagi. (2020, July 5). *Why choose MERN stack? - Aeologic - Medium*. Medium; Aeologic. https://medium.com/aeologic/why-choose-mern-stack-323b4d95e4ea

Gillis, A. S. (2017). *REST API (RESTful API)*. SearchAppArchitecture; TechTarget. https://searchapparchitecture.techtarget.com/definition/RESTful-API

Git. (2021). Git-Scm.com. https://git-scm.com/

Google Fonts. (2021). Google Fonts. https://fonts.google.com/specimen/Raleway

Google Fonts. (2021). Google Fonts. https://fonts.google.com/specimen/Montserrat?query=mo

Informed Publishing. (2011). *Critical Care ACLS Guide*. Google.com. https://play.google.com/store/apps/details?id=com.informedpublishing.CriticalCare&hl=en_US

Institution of Engineering and Technology. (2017, November 16). *Fig. 1: Human-computer interaction and related research fields*. ResearchGate; ResearchGate. https://www.researchgate.net/figure/Human-computer-interaction-and-related-research-fields_fig1_321147663

Jones, J. D. (2019). *Four React.JS Design Patterns You Need To Be Know About - Jon D Jones*. Jondjones.com. https://www.jondjones.com/frontend/react/design-patterns/four-react-js-design-patterns-you-need-to-be-know-about/

Joo, H. (2017). A Study on Understanding of UI and UX, and Understanding of Design According to User Interface Change. *International Journal of Applied Engineering Research*, *12*, 9931–9935. http://www.ripublication.com/ijaer17/ijaerv12n20_96.pdf

Kempema, J. M. (2009, August 28). *Pedi STAT*. App Store. https://apps.apple.com/us/app/pedi-stat/id327963391

Lars Vogel (c) 2007 - 2021 vogella GmbH. (2021). *Unit Testing with JUnit 5 - Tutorial*. Vogella.com. https://www.vogella.com/tutorials/JUnit/article.html

Le, P. (2020, May 12). *Human-centred design in hospital technology - UX Collective*. Medium; UX Collective. https://uxdesign.cc/human-centred-design-in-hospital-technology-b8f89910c0e3

Listening is everything. (2021). Spotify.com. https://www.spotify.com/us/home/

Low-fidelity vs. high-fidelity prototyping. (2018). Invisionapp.com. https://www.invisionapp.com/inside-design/low-fi-vs-hi-fi-prototyping/

Material Design. (2021). *Material Design*. Material Design. https://material.io/design/foundation-overview

Material Design. (2021). *Material Design*. Material Design. https://material.io/design/environment/elevation.html

Material Design. (2021). *Material Design*. Material Design. https://material.io/design/layout/responsive-layout-grid.html

Material-UI: A popular React UI framework. (2021). Material-Ui.com. https://material-ui.com/

Mayo Clinic - Mayo Clinic. (2021). Mayoclinic.org; https://www.mayoclinic.org/

MVC Framework - Introduction - Tutorialspoint. (2021). Tutorialspoint.com. https://www.tutorialspoint.com/mvc_framework/mvc_framework_introduction.htm

MVC Framework - Introduction - Tutorialspoint. (2021). Tutorialspoint.com. https://www.tutorialspoint.com/mvc_framework/mvc_framework_introduction.htm

MySQL. (2021). Mysql.com. https://www.mysql.com/

Ossus GmbH. (2019, February 11). *Eponyms.* App Store. https://apps.apple.com/us/app/eponyms/id286022402

Programming Fundamentals/Program Design - Wikibooks, open books for an open world. (2019). Wikibooks.org. https://en.wikibooks.org/wiki/Programming_Fundamentals/Program_Design#:~:text=Program%20design%20consists%20of%20the,Understanding%20the%20Program

Proto.io. (2017, January 20). *Why Wireframes Are Important in the Design Process.* Medium; Medium. https://protoio.medium.com/why-wireframes-are-important-in-the-design-process-de4e773e611

React – A JavaScript library for building user interfaces. (2020). Reactjs.org. https://reactjs.org/

Rungta, K. (2020). *What is Functional Testing? Types & Examples (Complete Tutorial).* Guru99.com; Guru99. https://www.guru99.com/functional-testing.html

Rungta, K. (2020). *What is System Testing? Types & Definition with Example*. Guru99.com; Guru99. https://www.guru99.com/system-testing.html#:~:text=SYSTEM%20TESTING%20is%20a%20level,a%20larger%20computer%2Dbased%20system

Saffer, D. (n.d.). *Creating Innovative Applications and Devices SeConD eDItIon designing for interaction*. http://www.sylviafredriksson.net/PCA/CD/IM2016/documents/readings/DSaffer_C1.pdf

Software Testing - Quick Guide - Tutorialspoint. (2021). Tutorialspoint.com. https://www.tutorialspoint.com/software_testing/software_testing_quick_guide.htm

Symptom Checker：概要 | Rakuten RapidAPI. (2019). Rakuten APIs. https://english.api.rakuten.net/priaid/api/symptom-checker/details

Symptomia. (2012). *Symptomia - App Store*. App Store. http://www.app-store.es/symptomia

Teo Yu Siang. (2020, July 25). *What is Interaction Design?* The Interaction Design Foundation; UX courses. https://www.interaction-design.org/literature/article/what-is-interaction-design

The Difference Between UX And UI Design - A Beginner's Guide. (2021). Careerfoundry.com. https://careerfoundry.com/en/blog/ux-design/the-difference-between-ux-and-ui-design-a-laymans-guide/

The Importance of Style Guides - Extractable. (2013, September 3). Extractable. https://www.extractable.com/insights/the-importance-of-style-guides/#:~:text=It%20provides%20the%20strategy%20behind,hired%20production%20teams%20and%20clients

User Testing: Why It Is Important and How to Do It Right. (2015, April 9). Build Blog by ThinkApps | Content on Entrepreneurship, Mobile Apps, Web Platforms and More. http://thinkapps.com/blog/design/user-testing-guide/

Visual Studio Code. (2016, April 14). Visualstudio.com. https://code.visualstudio.com/

Vue.js. (2021). Vuejs.org. https://vuejs.org/

Weber, J. (2018, May 24). *Why You Need User Testing (And How to Convince Others to Feel the Same)*. Medium; UX Planet. https://uxplanet.org/the-case-for-user-testing-87d82da3c19c#:~:text=User%20testing%20allows%20product%20teams,exposed%20to%20the%20larger%20market

What Are User Flows In User Experience (UX) Design? (2019). Careerfoundry.com. https://careerfoundry.com/en/blog/ux-design/what-are-user-flows/#:~:text=User%20flows%2C%20UX%20flows%2C%20or,through%20to%20the%20final%20interaction

What is a Flowchart. (2021). Lucidchart. https://www.lucidchart.com/pages/what-is-a-flowchart-tutorial

What is AGILE? - What is SCRUM? - Agile FAQ's | Cprime. (2019). Cprime. https://www.cprime.com/resources/what-is-agile-what-is-scrum/#:~:text=Scrum%20is%20a%20subset%20of,be%20consistent%20with%20the%20framework

What is an application architecture? (2021). Redhat.com. https://www.redhat.com/en/topics/cloud-native-apps/what-is-an-application-architecture

What is Human-Computer Interaction (HCI)? (2014). The Interaction Design Foundation; UX courses. https://www.interaction-design.org/literature/topics/human-computer-interaction

What is Regression Testing? Definition, Tools, Method, and Example. (2015, March 25). Softwaretestinghelp.com. https://www.softwaretestinghelp.com/regression-testing-tools-and-methods/

What is the MERN Stack? Introduction & Examples. (2021). MongoDB. https://www.mongodb.com/mern-stack

What is User Centered Design? (2021). The Interaction Design Foundation; UX courses. https://www.interaction-design.org/literature/topics/user-centered-design

What is User Centred Design? A guide to the processes involved | PRWD. (2017, August 29). PRWD. https://www.prwd.co.uk/blog/user-centred-design-process-overview/

Why Designers Shouldn't Neglect Mockups. (2015, March 19). Studio by UXPin. https://www.uxpin.com/studio/blog/designers-shouldnt-neglect-mockups/#:~:text=As%20a%20transitional%20phase%20between,final%20form%20of%20the%20product

Why is software testing necessary? (2019, July 26). Tryqa.com. http://tryqa.com/why-is-testing-necessary/#:~:text=Software%20Testing%20is%20necessary%20because,them%20are%20expensive%20or%20dangerous.&text=Software%20testing%20is%20really%20required,made%20during%20the%20development%20phases.

Wilson, M. (2013, January 31). *Infographic: The Intricate Anatomy Of UX Design*. Fast Company; Fast Company. https://www.fastcompany.com/1671735/infographic-the-intricate-anatomy-of-ux-design

9 Appendices

9.1 Appendix A – User Survey

This link shows the user survey that was used to determine requirements for the application.

https://docs.google.com/forms/d/e/1FAIpQLSe1v6F5m0WxB3olMbYhI79vczzk4HdzpdHyelduWYUb8L7rVg/viewform?usp=sf_link

9.2 Appendix B – Interactive Prototype

This link contains the interactive user interface prototype with InVision.

https://projects.invisionapp.com/prototype/Mockups-Prototypes-Mobile-1st-ckle7j6d100agnr01avw7hbxi/play/5a442564

9.3 Appendix C – Code Repository

GitHub repository with all the codes for Auxillia Diagnostics.

N00172468/Y4_RP_MED_DIAGNOSTIC_TOOL_PROJECT: This final year project is based on a Medical Diagnostic Tool. It's a web-app app that serves as a reference tool for medical students, doctors/nurses, general practitioners and/or emergency workers. It would help medical students and staffs to look up symptoms, determine possible causes and recommend appropriate diagnosis. Project is developed using React.js, Material UI and MERN Stack. (github.com)

9.4 Appendix D – Diagrams

This link directs to the Google Drive folder that contains all of the diagrams ranging from the technical diagrams all the way through the Hi-Fi prototypes.

https://drive.google.com/drive/folders/1Pvw4X_H40wA7HWpPQL_l0m2o4uL7zyDC?usp=sharing

9.5 Appendix E – Interviews

Interview Results with ICU Registered Nurse:

Interviewer: “What field, department or ward do you work in?”

Interviewee: “I work in the ICU ward.”

Interviewer: “How long have you been working in that field?”

Interviewee: “Only a year so far.”

Interviewer: “If you had to research a certain information, how would you look it up? For example, while researching for particular symptoms, would you use the internet, books, research papers, etc.”

Interviewee: “By research papers and continuously training my skills.”

Interviewer: “Once you’ve gained the information that you need, how do you record that information? For example, by typing up a report, writing it down on a notebook, bookmarking the webpage, etc.”

Interviewee: “By bookmarking the webpage and by making a PowerPoint presentation.”

Interviewer: “How do you organise the information that you have recorded? For example, do you use sticky notes then place them on your medical clipboard, do you use flashcards to learn off the information, etc.”

Interviewee: “Electronically as in by typing it as notes or saving it as a bookmark in my browser.”

Interviewer: “If you were to find a ground-breaking discovery, let’s say you identified a new symptom or found an alternative way to cure an existing disease, would you publicise your newfound knowledge? If so, how would you announce it to the general public?”

Interviewee: “Yes, definitely! It would be helpful so it can update the seat guidelines.”

Interviewer: “If you were to use an app to help you find the research materials that you need quickly and reliably, but have questions about a certain information, would adding a comment section to communicate with other medical staffs around the world about that topic be helpful for you?”

Interviewee: “I believe so, yeah. It would be helpful to share knowledge with colleagues.”

Interview Results with former Medical Chemistry Student:

Interviewer: “What field, department or ward do you want to work in?”

Interviewee: “I want to be in Pharmaceutical Science.”

Interviewer: “When it comes to studying, how do you look up the information that you need? For example, researching symptoms, possible causes of those symptoms and how to determine the appropriate diagnosis.”

Interviewee: “Google, eBooks, ScienceDirect and lab reports. But mainly ScienceDirect.”

Interviewer: “Once you’ve gained the information that you need, how do you record that information? For example, by typing up a report, writing it down on a notebook, bookmarking the webpage, etc.”

Interviewee: “Bookmarking, writing reports and jotting down notes.”

Interviewer: “How do you organise the information that you have recorded? For example, do you use sticky notes then place them on your wall, do you use flashcards to learn off the information, etc.”

Interviewee: “By word doc. And typing down notes.”

Interviewer: “If you were to use an app to help you find the research materials that you need quickly and reliably, but have questions about a certain information, would adding a comment section to communicate with other students or medical staffs around the world about that topic be helpful for you?”

Interviewee: “Yes, definitely!”

9.6 Appendix F – Test Preparation for User Testing

Introduction:

- “Thank you for taking time to help us.”
- “I’ll/we’ll be showing you a prototype of a medical diagnostic and reference application.”
- “The reason I’m/we’re showing you this is to get feedback before the team builds the real version.”
- “Don’t worry about breaking anything. I/we encourage you to do so as it will help us out with fixing those unknown bugs.”
- “I’ll/we’ll be providing you a few tasks to roleplay.”
- “Please think out loud as you do the tasks as well as using the prototype. Do not be afraid to say literally anything and everything. There are no wrong answers.”

Consent Form:

In this usability test, you will be asked to perform certain tasks on a computer. We will also conduct an interview with you regarding the tasks you perform.

Participation in this usability test is voluntary. All information will remain strictly confidential. The descriptions and findings may be used to help improve the *Auxillia Diagnostics* application. However, at no time will your name or any other identification be used. You may withdraw your consent to the experiment and stop participation at any time.

If you have any questions after today, please contact John Doe at john.doe@testing.ie.

I have read and understood the information on this form and had all of my questions answered

_____ _____

Subject’s Signature Date

Pre-test Questionnaire:

- What field do you work in/want to work in?
- How do you look up the information that you would need to research?
- How do you record the obtained information?

Post-test Questionnaire:

- What do you think about the application?
- Was it simple and satisfying to use?
- What did not appeal to you?
- What would you like to be improved within this application?
- Are there other features you would like to have within this application?