|  |  |
| --- | --- |
|  | Mobile Application Delivering Seamless User Experience |
|  |  |
|  | Ogundiran Jamie  CE301 capstone project 2022~2023  4/28/23 |

# Abstract

The use of smartphones has increased exponentially in recent years, due to the accessibility and the convenience of mobile applications. Although, there are about 3.04 million mobile applications globally (alvin, 2021), a mobile app that satisfies the end-user and the client is rare to find. The task of satisfying the user needs is known to be a difficult task to accomplish for the developer. Comprehending the underlying problems, this project aims to create a mobile application that is engineered with a user-centered design approach to provide a seamless experience for both customers and employers. This is achieved by analyzing the data collected through user interview, use cases and competitive audit.

The mobile application which is iOS-based is a nail-salon appointment booking application, the main concept of the app is that it has a live interactive booking system, a simple database manipulation system for the employer/admin for the employers with no technical background and a live salon statistics system that displays status of the energy consumption rate and the revenue of the company. Moreover, the project includes other extra features such as social media integration and integrated geolocation function to satisfy the user needs in which we will discuss in this paper.

The mobile application is developed using the react native framework with Firebase Fire store as the database management system.

Overall, the documentation and mobile application produced in this project are highly valuable resources for developers seeking to create user-centric mobile applications. And it represents an important step towards enhancing the overall quality of mobile applications.

# 1.Introduction

The advent of smartphones has led to a dramatic increase in their usage, with sales of smartphones to end-users surging from 296 million in 2010 to an astonishing 1433 million in 2021 (O'Dea, 2022). As the demand for smartphones continues to rise, so does the need for high-quality mobile applications. Nowadays, tasks that used to require users to call businesses and make reservations can be easily completed through mobile apps. However, creating a mobile app that satisfies the user's needs and preferences can be a daunting task, requiring developers to conduct extensive UX and UI research.

In this project, I will be conducting a thorough UI and UX research to create an innovative and user-friendly mobile app for a salon business. The app will include features such as a live interactive booking system, a simple database manipulation system for non-technical employers/administrators, and a live salon statistics system that provides real-time data on energy consumption and revenue. With these features, customers can easily book appointments, while salon owners can efficiently manage their business operations and make informed decisions based on real-time data.

Our client, Kaoru Ogundiran, owns a popular nail salon named "Sharon" in Tokyo, Japan, which is highly sought after by tourists. The salon has gained immense popularity through social media, and currently utilizes a third-party website called "Hot Pepper Beauty" for menu viewing, contact details, and appointment bookings. However, due to the language barrier, tourists often face difficulties using the website. Additionally, with the lifting of COVID-19 restrictions, our client foresees an increased demand for a booking application written in English. Hence, our project aims to create a user-friendly mobile application in English that will allow tourists to easily book appointments and access salon information, thus enhancing the overall customer experience at the salon.

# 2.Related Work

## 2.1 Software Framework

There are several software options available for mobile app development, each with its unique benefits. However, selecting the best framework for a particular project can be challenging since there are dozens of frameworks to choose from. Some of the popular frameworks include React Native, Flutter, Swiftic, Ionic, and Jetpack Compose. In this section, we will be examining the performance of React Native, Flutter, and Jetpack Compose to determine the most suitable framework for our project.

### 2.1.2 React Native

React Native is a cross-platform framework used for developing mobile applications, which was first released by Facebook in 2015 (Wu, 2018). React Native is written in JavaScript, but it allows for the rendering of views and access to native hardware such as camera and storage by using underlying native interfaces, making it different from hybrid or HTML5 applications. One of the key benefits of React Native is its open-source nature and strong community support, with contributions from Facebook developers, individuals, and even companies like Microsoft and Samsung.

React Native uses JSX, a special syntax extension of JavaScript, to describe how the user interface is displayed. When the application is built, the JSX is compiled into regular JavaScript code. There are two main data models in React Native: Props and State. Props are set externally and used to customize the component, while State is set internally and used to initialize values. The use of Props and State allows for efficient and effective component reusability in React Native applications.

### 2.1.2 Flutter

Flutter is another cross-platform framework used for developing mobile applications, which was released by Google in 2016 (Wu, 2018). Unlike React Native, Flutter applications can run on Android, iOS, and Fuchsia. One of the key features of Flutter is its high-performance rendering engine, which renders every view component using its own engine instead of relying on web views, as is the case with React Native.

Every application in Flutter is written in Dart, a programming language developed and maintained by Google. Since Dart was developed as a replacement and successor to JavaScript, it implements most of the characteristics of JavaScript's next standard (ES7) (Wu, 2018). In terms of data structure, Flutter uses Widgets instead of Props and State. Essentially, the application is a collection of widgets that interact with each other. Widgets contain features such as creating input text boxes, buttons, and more.

One of the advantages of Flutter over React Native is its rapid development capabilities, thanks to the availability of widgets, including those for animation and gesture detection. Additionally, Flutter has a hot reload feature, which allows for quick updates of the application state when the save button is clicked.

### 2.1.3 Jetpack Compose

Jetpack Compose is a modern UI toolkit developed by Google for creating native Android applications using the Kotlin programming language. The first stable public version of Jetpack Compose was released in July 2021 (Soininen, 2021). Jetpack Compose is designed to simplify the process of building complex and responsive user interfaces for Android apps. One of the key benefits of Jetpack Compose is that it allows developers to create UI components using a declarative programming model, which makes it easier to write and maintain code.

Unlike some of the other popular cross-platform mobile app development frameworks, such as React Native and Flutter, Jetpack Compose is focused solely on the Android platform. This means that it doesn't offer cross-platform support for iOS or other mobile operating systems. However, for developers who are specifically focused on building Android apps, Jetpack Compose can be a powerful tool for creating high-quality user interfaces quickly and efficiently.

### 2.1.4 Performance Comparison

There is a study by Wenhao Wu comparing React Native and Flutter, which are the two leading frameworks in mobile app development. Wenhao compared the two frameworks by writing a simple TV show application in React Native and rewriting it in Flutter. Upon using each framework, he discovered that both Flutter and React Native perform well in terms of scrolling, with the average frames per second (fps) while scrolling being over 60 fps (Wu, 2018). However, when the list being scrolled through is dense, React Native is said to have a significant drop in fps, whereas Flutter remains stable. Wenhao also compared the speed of the input and output system (I/O) of React Native and Flutter and found that React Native has an advantage in both the average time and single time consumption. The measurement was calculated by determining the time of an obtained list of epochs from opening a file to successfully writing on a file.

From the comparison above, we can conclude that React Native is a better framework than Flutter; however, there are many app development frameworks other than Flutter. In a study paper produced by Visa Soininen, the author compares the performance between React Native and Jetpack Compose. He developed two applications using React Native and Jetpack Compose, which had features such as user authentication (login/sign up), bottom navigation bar, and item listing. When testing the performance of each application, he evaluated the compiling speed and rendering speed. From the comparison, the author discovered that React Native has better performance with both compiling speed and rendering speed. The initial time taken to bundle the application and the subsequent time after the initial bundle were used for the comparison. After taking an average of five results, he found that React Native obtained an initial bundle time and a subsequent build time of 21.5 seconds and 54 milliseconds, whereas Jetpack Compose obtained an initial bundle time and a subsequent build time of 22.5 seconds and 1,112 milliseconds (Soininen, 2021). Furthermore, the rendering speed of each framework was obtained by utilizing the button components in each framework to render a new object. After taking an average of five results, he found a rendering speed of 450 milliseconds for Jetpack Compose and 371 milliseconds for React Native.

In conclusion, the data gained from the research papers above suggests that React Native is the optimal mobile development framework to be used in this project.

## 2.2 Database

Database is a structured set of data stored in the computer, the system that manages is called Database Management System (DBMS) (Margaretha Ohyver, Jurike V. Moniaga, Iwa Sungkawa, Bonifasius Edwin Subagyo, Ian Argus Chandra, 2019). The primary purpose of a DBMS is to simplify the process of writing, fetching, and updating data for the user. As a result, users often prefer DBMS software with a fast response time, as this can significantly improve the overall efficiency of the application. However, the response time of a DBMS can be influenced by various factors, including the development language and application framework used. Choosing the right DBMS software with the fastest response time for your specific application can therefore be challenging. In this section I will be researching two DBMS, firebase, and MySQL database.

### 2.2.1 Firebase

Firebase, also known as Firebase Realtime Database, is a mobile and web application development platform developed by Google. It provides tools for developing high-quality mobile and web applications, including real-time databases, authentication, messaging, hosting, and more. One of the main features is Firebase Realtime Database, which, as the name implies, updates any data on the Google server immediately, making it available everywhere. (Margaretha Ohyver, Jurike V. Moniaga, Iwa Sungkawa, Bonifasius Edwin Subagyo, Ian Argus Chandra, 2019).

### 2.2.2 MySQL

MySQL is an open-source relational database management system (RDBMS) used to store and manage structured data. Originally created by the Swedish company MySQL AB, it was later acquired by Oracle Corporation. The database comprises multiple tables, each with several columns known as fields. To establish an integrated database system in MySQL, it is essential to maintain the relationship between tables. This can be achieved by using a primary key in one table and a foreign key in another. (Margaretha Ohyver, Jurike V. Moniaga, Iwa Sungkawa, Bonifasius Edwin Subagyo, Ian Argus Chandra, 2019)

### 2.2.3 Performance Comparison

To determine the optimal DBMS for our application, I analysed research conducted by a group of students in Indonesia. The research compared the performance of two databases, Firebase Realtime Database and MySQL Database, in a mobile application designed for tracking toddlers' daily nutritional needs. The database response time was measured by recording the time interval for the CRUD operation, which is the basic function for retrieving and returning data from a database. The CRUD operation was performed 50 times, and the data obtained was plotted in a graph using the Wilcoxon Signed-Rank test. The test results revealed that the Firebase Realtime Database outperformed the MySQL Database in all CRUD operations, including CREATE, READ, UPDATE, and DELETE operations. Based on these findings, we can conclude that the Firebase Realtime Database is more optimal for mobile application development.

# 3.USER RESEARCH

### 3.1.1INTERVIEW

The interview was conducted with two individuals of different ages, covering a wide range of users.

Interviewer: Jamie Ogundiran Interviewee: Kaoru Mishimaki

Q. Could you give me a brief introduction of yourself? A: My name is Kaoru Mishimaki. I was born on October 13th, 1976, and I manage a nail salon.

Q. What salon app do you use for booking appointments? A: I use an app called Hot Pepper Beauty, which is a salon booking app based in Japan.

Q. Tell us about your experience with Hot Pepper Beauty. A: Overall, my experience with Hot Pepper Beauty has been great. The app recommends a salon to me based on the hairstyle I want. However, at times, the hairstyle on the salon might not be accurate.

Q. Have you used Hot Pepper Beauty to book an appointment for your nails? If so, how was your experience? A: Yes, I have. The experience was good, and I did not have any issues with the booking process. The app also has a feature that allows me to sort the nearest salon based on my current location.

Q. What app do you use for managing salon appointments? A: I use an app called Salon Board.

Q. What is your favourite feature of Salon Board? A: The app has a feature that calculates the revenue of the salon, which is very helpful. However, I wish there was a way to calculate the energy prices in the app as well.

Interviewer: Jamie Ogundiran Interviewee: Sarah Ogundiran

Q. Could you give me a brief introduction of yourself? A: I am interested in management and economics.

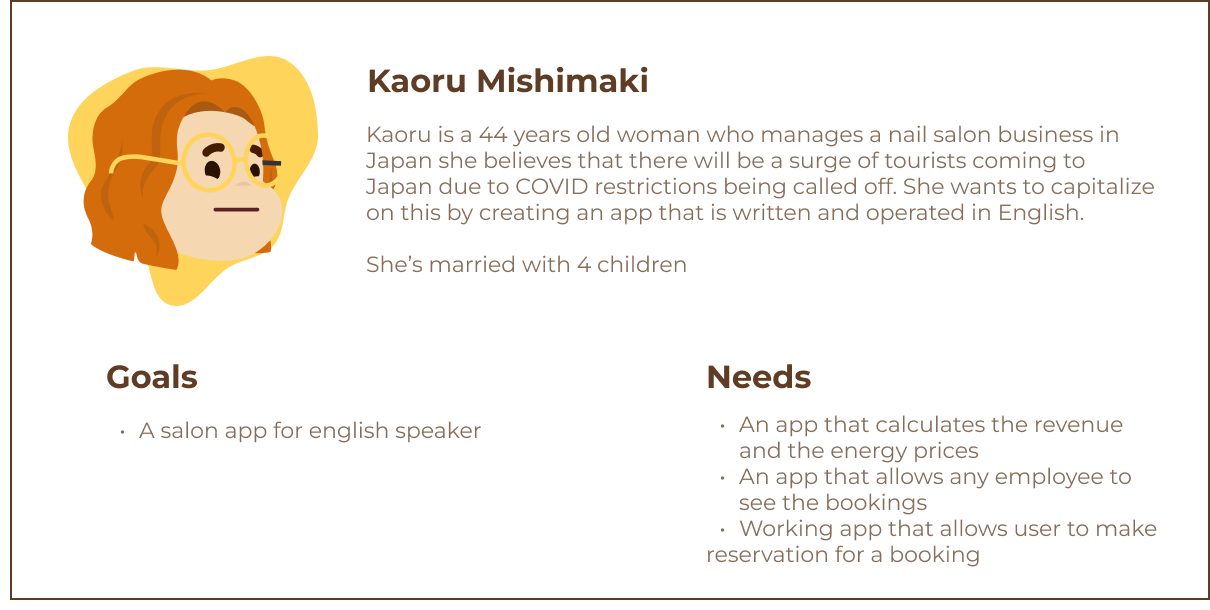
Q. What salon app do you use for booking appointments? A: I use Simply Hair Salon.

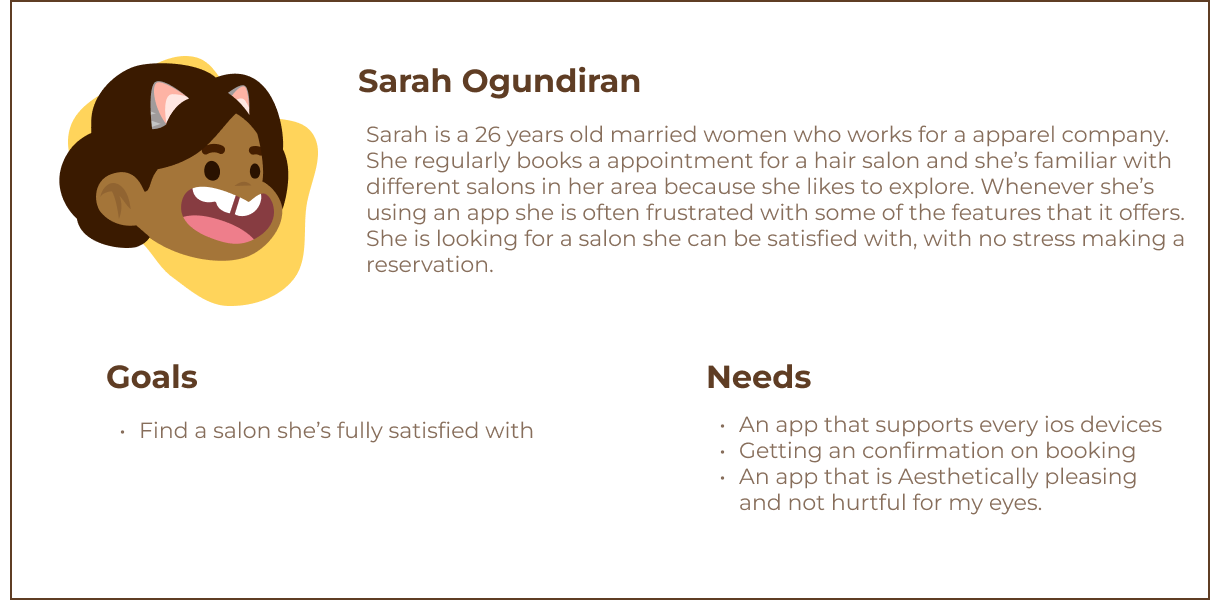
Q. Tell us about your experience with Simply Hair Salon. A: The app looked old and was not supported on my device. The calendar did not fit my phone, and I had to zoom in to book an appointment. I did not receive an email confirmation, so I had to take a screenshot. The app theme was not well-done, with a black background.

Q. What is your favourite feature of Simply Hair Salon? A: My favourite feature is the gallery page, which gave me a direct insight into what my nails would look like.

#### USER PERSONA

I have created a user persona of the two individuals interviewed. The user persona captures the needs, motivations, and behaviours of the targeted audience.





#### Analysis

By conducting a user interview and creating a user persona, I was able to gain valuable insights into the needs and preferences of potential end-users of the app. Specifically, I discovered that the client, who serves as the manager of the salon, is primarily interested in an app that facilitates simple reservations and provides a clear overview of the financial performance of the salon. Additionally, the client is interested in an app that enables employees to check the status of bookings and appointments.

On the other hand, the customer's needs were focused on the usability and accessibility of the app. They expressed a desire for an app that supports multiple devices and provides confirmation of bookings or appointments. This insight highlights the importance of creating an app that is not only functional but also user-friendly and accessible to a wide range of users.

### 3.1.3 COMPETITIVE AUDIT

In app development, competitive analysis plays a crucial role in the development cycle. Competitive analysis allows the programmer to understand the market, forecast market potential, track competitor pricing and product and many other. For my program I have decided to conduct competitive analysis on two nail salon company, Young LDN and Townhouse nail salon. They are London based nail salon which offers services for hair, nail, and skin.

#### Young LDN nail salon:

#### 

##### Value Proposition:

The app Young LDN offers customers the convenience of booking appointments for various services such as manicures, pedicures, facials, IPL, laser hair removal, anti-ageing treatments, and more. Additionally, the app provides a feature that enables customers to purchase products directly from the app, including makeup tools, diffusers, skin care products, and nail care products.

The primary colour of Young LDN is a deep and rich shade of purple, which is often associated with luxury, sophistication, and creativity. This colour choice reflects the high-end nature of the services provided by the salon. In addition to purple, the branding incorporates shades of white and gray, creating a sense of balance and harmony. These neutral colours help to offset the boldness of the purple, creating a cohesive and elegant aesthetic for the app. Overall, the app's branding conveys a sense of luxury, quality, and professionalism, appealing to customers seeking high-end beauty services and products.

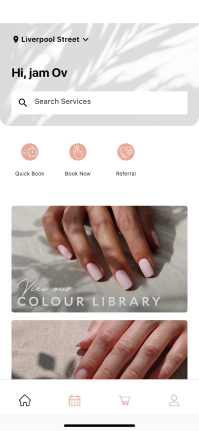
##### Customers Are Saying:

The feedbacks on this app are mostly positive with an average score being 4.9 out of 5 with 1000+ reviews on classpass (Review website). many customers stated how, easy and smooth the appointment process were, and they had no trouble with any of the features provided.

##### Pros & Cons

* Pros:
  + Great use of colours to express their brand identity
  + Easy to navigate through the appointment process
* Cons:
  + The website is very clamped together

#### Townhouse nail salon:

##### Value Proposition:

Like YoungLDN, “My Townhouse” salon provides the customer with the convenience of booking appointments and purchasing products directly from the app. The salon offers similar services such as manicures, pedicures, brows, express facials and sells items similar to YoungLDN such as make up tools, diffusers, skin care products and nail care products. The difference between the two salon is that My Townhouse has a mobile application that lets the user book an appointment and purchase products.

“My Townhouse” salon has carefully chosen a colour palette of white and pink, which perfectly reflects its target audience of older women. The elegant and sophisticated combination of these colours exudes femininity, delicacy, and romance, creating a calm and serene atmosphere in the salon. The use of white invokes feelings of purity, simplicity, and clarity, while the soft pink exudes a gentle and soothing vibe, representing kindness, empathy, and compassion.

##### Customers Are Saying:

The feedback on this app is mostly great with the app overall rating being 4.8 out of 5. Few negative comments are regarding the bugs that happens using the app.

##### Pros & Cons

* Pros:
  + Great use of colours to express their brand identity
  + Has a search system on the app version
  + Minimal design and easy to follow
* Cons:
  + None

#### Analysis

Through conducting a competitive audit on two salon websites/apps, I have gained valuable insights into what aspects of these applications are user-friendly and what areas could be improved upon. One of the most critical takeaways was the importance of using a colour palette that reflects the target audience, as well as implementing easy navigation between screens with clear instructions and minimal layouts and structures. These elements are crucial in building a user-centred application that is both easy to use and visually appealing.

Additionally, I was particularly impressed with the layout that My Townhouse had implemented in their app version of the website. As a result, I plan to incorporate this layout and structure as the basis for my own app design, allowing me to provide a seamless user experience that is both intuitive and functional.

# 4.Methodology

## 4.1 SYSTEM DIAGRAM

A system diagram is a graphical representation of a system that shows the various components and their interactions. It provides a visual overview of the system and can be used to understand the system's structure, behaviour, and relationships with other systems [Gharajedaghi, 2011]. The system diagram of the mobile application is given below:

Diagram

Description automatically generated

The diagram above displays the system diagram of the mobile application, it consists of 4 main parts, Login/Register, admin app, customer app and a database system. The general functionality of the register/login page is authentication and creation of the user in the database (firestore). The user is only able to access the admin app or the Customer app only if they are authorized through the database. The database I am using for authentication and management is Firebase. Firebase allows for an easy authentication allowing smooth transition between the login page and the Homepages.

The admin app and customer app consist of main screens where users can interact with. The customer app contains screens such as home page, gallery page, profile page, booking age and contact page. On the other hand, the admin app contains screens such as home page, add menu page, revenue page and energy consumption page. In customer app, the homepage is the initial screen of the app and then it branches to other pages, The booking page allows the user to book an appointment, the contact page allows user to interact with the employee of the salon, gallery page displays a pictures of nail designs and profile page displays general user details. In the admin side of the app, similarly the homepage is the initial screen of the app, add menu page allows for addition of new menu or deletion of current menu’s, revenue page displays the revenue for the month and the energy consumption displays the energy consumption of the company for the month.

The main functionality of the database system is authentication and database management, The functionality is used during the login/register phase, the admin app and the customer app.

Moreover, the screen pages in customer app and the admin app have a logout feature which navigates the user back to the login screen.

## 4.2Login/Register

Diagram

Description automatically generated

The system diagram displays the flow structure of the Login and register process of the customer and the admin. The initial page is the login page, where the user has the option to login/register as a customer or login as an admin. Once the user is initiates the login process, the firebase authenticates the user and if authenticated, the user is navigated to the homepage od the application. If the user is not authenticated an error message will display. The homepage differs depending on if the user is a customer or an admin.

The register process is different from the login page, since it includes feature, such as creating a new user detail and email verification. When the user initiates the registration process in the register page, the database sends a verification email to the email provided by the user. User authentication details are only created if the email address provided is verified by the user. Once the verification and creation of user is completed, it navigates the user to the home page of the application. This process is only available for the customer, meaning that admin cannot create new user authentication for admin login.

## 4.3Home Page (Customer App)

Diagram

Description automatically generated

The system diagram displays the flow structure of the homepage, it includes 5 components, drawer navigator, gallery page, contact page, booking page and tab navigator.

## 4.4 Gallery Page

Diagram

Description automatically generated

The system diagram displays the flow structure of the gallery page, it includes 5 components, tab navigator, main gallery page and the drawer navigator.

## 4.5 Contact Page

Diagram

Description automatically generated

The system diagram displays the flow structure of the contact page, it has three features, email us, phone number and location. Upon the button pressed email us directs the user out of the nail salon app into the Gmail app, the user is then able to contact the company, moreover upon pressing the phone number button the user is then given an option to call the company phone number, finally the location button navigates the user to google map with the location of the nail salon already given.

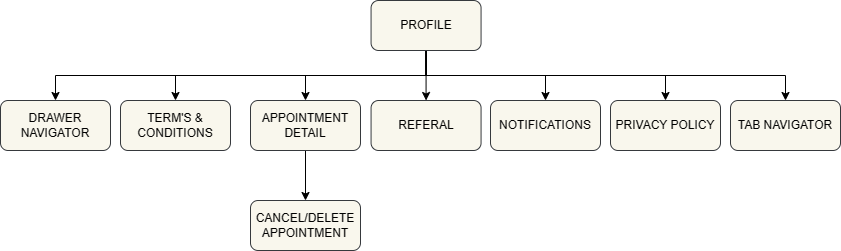
## 4.6 Booking Page

Diagram

Description automatically generated

The flow diagram of the booking page consists of four stages, booking page for the user to select a service from the menu, booking page for the user to select the date and time, booking page for the user to confirm the appointment details. In booking page (menu), the user will need to have a service selected in the menu to proceed to booking page (date). Similarly in booking page (date) the date and time are needed to select to proceed to the booking confirm page. After the details are confirmed, the confirm page navigates the user back to the homepage.

## 4.7 Profile Page



The system diagram above displays the flow structure of the profile page in customer app. The profile page includes 8 options, drawer navigator, terms & conditions, appointment detail, referral, notifications, privacy policy and tab navigator. Terms & conditions directs the user to the terms & condition of the salon, appointment detail displays the status of the booked appointments and allows cancellation of the appointment, referral allows the user to share the app on other platform, notification directs the user to the notification setting of the application and privacy policy displays the privacy policy of the app.

## 4.8 Home Page (Admin)

Diagram

Description automatically generated

The system diagram above displays the home page of the admin side app. It includes states such as drawer navigator, viewing booked appointment and cancel booked appointment. The homepage allows the admin to view every booked appointment detail and also delete booked appointment.

## 4.9 Add Menu Page (Admin)

Diagram

Description automatically generated

The system diagram above displays the add menu page, it includes features add menu and delete menu. The add menu adds a new menu/course into the database which would then be rendered to the customer app, similarly the delete menu deletes an existing menu.

## 4.10 Revenue Page (Admin)

Diagram

Description automatically generated

The system diagram above displays the revenue page, it includes the revenue main page and a revenue history state. The revenue main page is where the details of the revenue made for the following month is shown, moreover the history of completed booked appointment will also be displayed. The revenue main page includes a button which directs the user to a revenue history page. Revenue history page displays the history of total revenue made in each month.

## 4.11 Energy Consumption Page (Admin)

Diagram

Description automatically generated

The system diagram above displays the energy consumption page, it includes the energy consumption main page and an energy consumption history state. The energy consumption main page is where the details of the energy consumption price for the following month is shown, moreover, the predicted energy consumption rate of the month is displayed as well as the detailed energy usage of the salon. The main page includes a button which directs the user to an energy consumption price history page. The page displays the history of cost in energy for the past months.

# 5.Design

The design of a mobile application plays a critical role in its success. It's the first point of contact between the user and the application, and it's often what draws them in and keeps them engaged. A well-designed mobile application can increase user engagement, promote user loyalty, and enhance the user experience. On the other hand, a poorly designed application can lead to user frustration, negative feedback, and ultimately, low adoption rates. In this section, we will be creating the prototypes, wireframe, and the mock-ups of the application with the help of the information obtained from user research and system diagram.

The wireframe and the mock-ups were created using a design software called Figma. Which is a web-based graphics editing and interface design tool used for creating user interfaces, prototypes, and icons.

## 5.1Wireframe

A wireframe is a visual representation of a mobile application’s layout, structure, and functionality. It is a simplified, low-fidelity design that is used to communicate the basic layout and user interface of an application.

The diagram below displays the wireframe of our application:

Graphical user interface

Description automatically generated

## 5.2MockUp

Graphical user interface, application

Description automatically generated

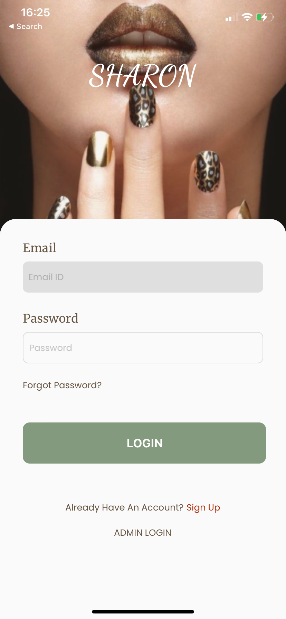
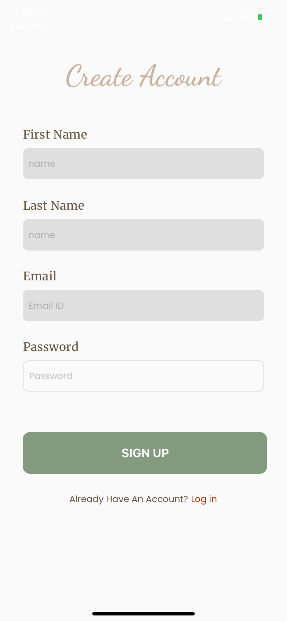
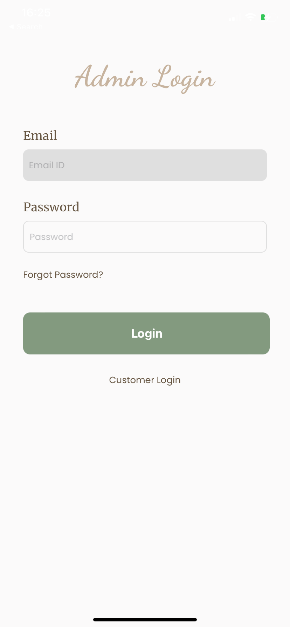
The Mock-up above displays the user interface design of the main components of the application. The main colour palette of the application is light brown as the primary colour, olive as the secondary colour, brown as the text colour, white as the background colour, white as the secondary text colour and grey for the icon colour. The main font is Merriweather, and the sub font is Poppins. The design is likely to shift due to my inexperience of the application framework (React-Native).

# 6.Implementation

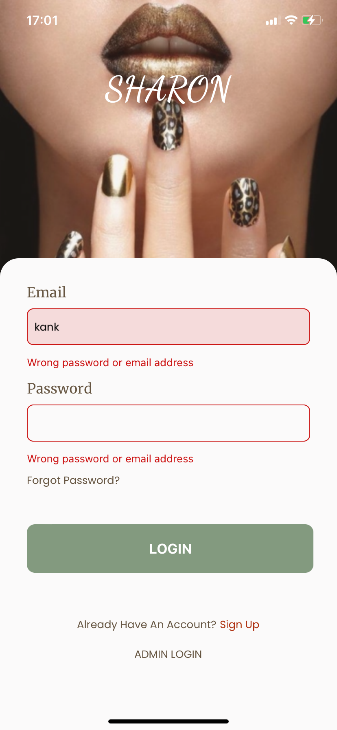
From the research papers analysed in the related section, we can conclude that Firebase and React Native is the optimal software for my application. Hence, I will be using React Native framework with JavaScript and Firebase as for the front-end and back-end in developing this mobile application.

## 6.1 Final Product

### 6.1.1 Signup/Login page

Here is the user interface of the final draft of the signup/login screens, As defined in the system diagram of the methodology section. The user is introduced to the signup/login page first. The main functionality of the pages in these sections are user authentication, user creation, email verification and password resetting. The user authentication is done using the singInWithPasswordAndEmail library from firebase auth, As stated in the name of the library the user is authenticated based on its email and password. When the authentication is wrong, the program displays an error message on the console, moreover the program indicates the user for authentication errors by outlining the input fields in red with a message. Moreover, the admin is only able to login to the admin login with the use of the authentication details set by the admin.

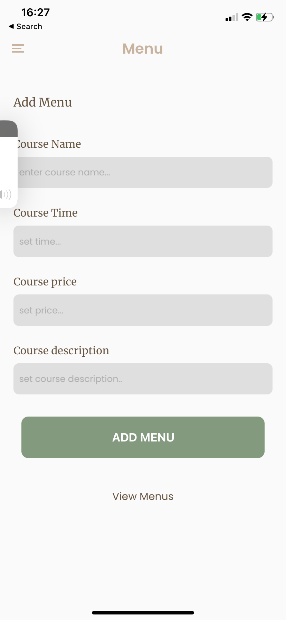
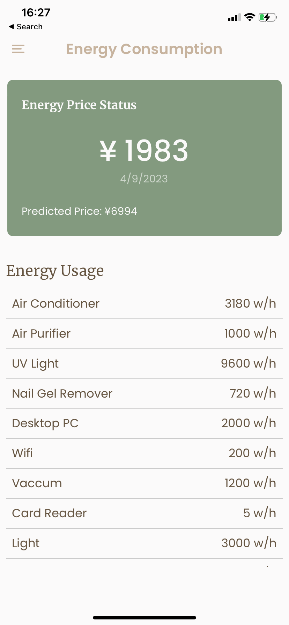
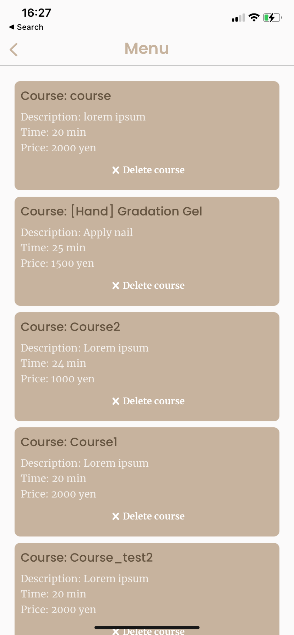


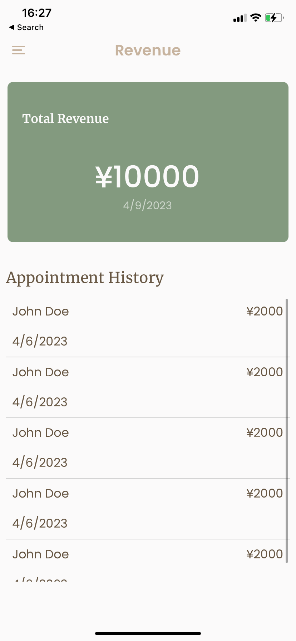
Another functionality, user creation, is done in sign up page where a new user is created in the firebase auth database. The user is created with the createUserWithEmailAndPassword library from firebase auth. As stated in the library’s name the user is created with Email and Password, upon creation uid (Unique identifier) is created in the firebase auth database.

Lastly, password resetting, which is a function that lets user to reset their password to a new password. This is achieved with the use of a built-in function in the firebase authentication. is done using the built-in function in firebase auth called sendPasswordResetEmail.

The navigation between the login page and the sign-up page is done using a react native library called stack navigator.

### 6.1.2Admin screens



The admin-exclusive screens depicted above showcase the comprehensive functionality of the application's backend. Accessible only by authorized admins via a secure login page, these screens provide a host of critical features, including appointment and course management, energy pricing information, and real-time revenue tracking.

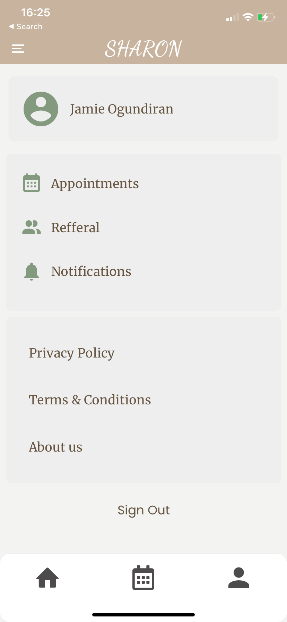
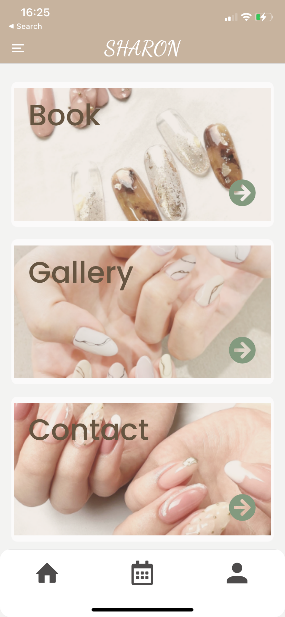
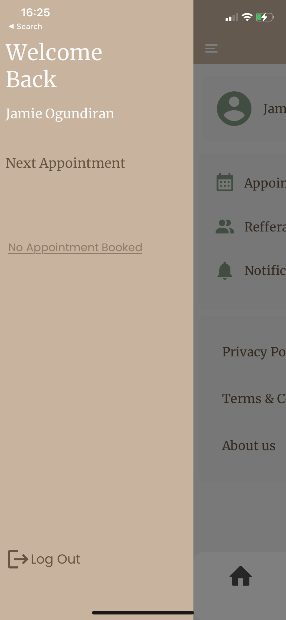
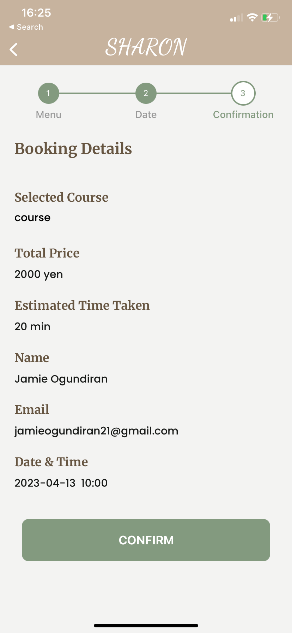
The appointment deletion and course deletion/addition features leverage powerful Firebase Firestore libraries, including collection, getDocs, deleteDoc, doc, and addDoc, to streamline document management processes. These libraries enable admins to quickly and easily add, delete, or modify critical appointment and course data, providing them with unparalleled control and flexibility.

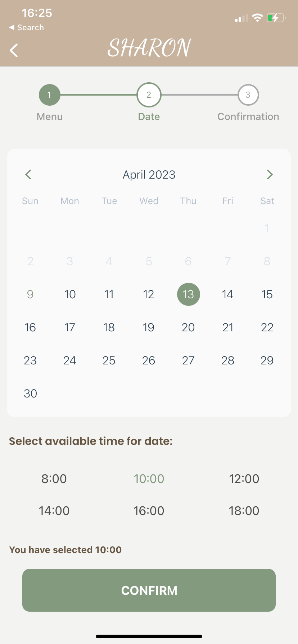
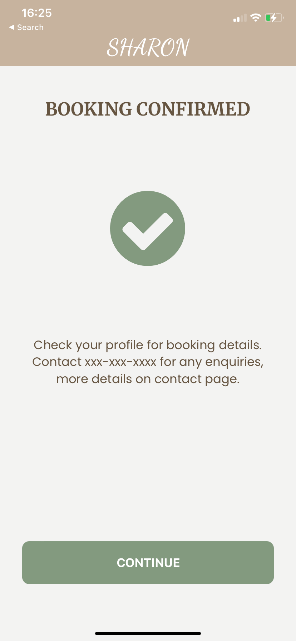
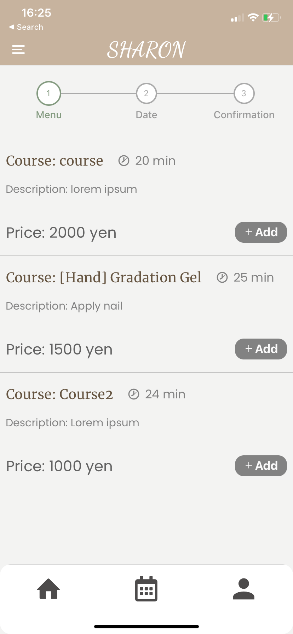
In addition, the energy consumption page offers a unique and intuitive approach to pricing, with dynamic calculations based on energy usage and electricity rates in Japan. By leveraging advanced mathematical algorithms, the application is able to precisely convert energy usage into kilowatt-hours and accurately determine the associated costs, providing users with unparalleled accuracy and transparency.

Furthermore, the navigation between the screens in admin pages are done using a react native library called drawer navigator.

Finally, the revenue tracking feature enables admins to gain a real-time view of total revenue including vat, providing them with valuable insights into business performance and financial health. By aggregating data from every past appointment, this feature provides a comprehensive overview of the business's performance, allowing admins to make informed decisions and optimize operations for greater success.

### 6.1.3 Customer Screen

The screens above display every page of the customer side of the application. The screens are accessible by the any customers if the user is authenticated, and the user can login through the customer login page in the signup/Login screens. The main functionality of this section of the app is live booking status and appointment booking. Similarly, to the admin side of the application, the live booking status is displayed using the firebase firestore library called collection and getDocs. The documents are fetched from the database and displayed on the screen using a flatlist library from react native. The booking process retrieves the data inputted by the user and store it in the database. The calendar library which renders calendar to the screen is used for allowing user to pick date and time for the appointment. Moreover, the data obtained from the booking process is stored into the database with the use of addDocs library.

Not only does the application offer essential functionalities, but it also boasts a range of supplementary features that enhance the user experience. These include a step indicator to guide users seamlessly through the app, convenient options for contacting the salon, a hassle-free account logout feature, and customizable notification settings to ensure users stay up to date with the latest information.

* Step indicator: The step indicator implemented with a library react native step indicator is to guide users through multi-step processes or workflows. They provide a clear visual representation of the user's progress and the number of steps remaining, which can help to increase engagement and reduce user frustration
* Contacting the salon: The contact page includes of locating the salon on the map, calling the salon from the app, and emailing the salon. These three features are implemented with the use of react native email, react native call and react native map library.
* Logout: The logout feature logouts the user from the authenticated account, the feature is implemented with the built in signout function from the firebase auth library.
* Notification: the notification feature directs the user to the device notification setting screen. This feature is implemented with the notification from react native notification library.
* Calendar: The calendar implemented with a react native library “react-native-calendar” offers the user an easy-to-use interactive calendar where the user is able to pick date and time for the booking.
* Booking live status:

### 6.1.4 Data Structure (Back-End)

Diagram

Description automatically generated

The data structure above which is stored in firebase firestore showcases the database system of the application. The database is split into two main data storage system, firestore and firebase Auth. Firestore manages data such as appointment data, user data, energy prices, and revenue. Whereas the firebase Auth stores the password, email and the uid (unique identifier) of of the user, the data in firebase Auth is used for user authentication.

# 7.Project Management

## 7.1 Project Scope

The main goal of the mobile application is to design and develop a mobile application that allows live appointment booking, tracking the live statistics of the salon and dynamic data manipulation for the admin. The live appointment booking includes a dynamic booking system that enables users to easily make multiple bookings for various services, the live statistic of the salon includes the statistics of the energy consumption price for the month and the revenue made so far and the dynamic data manipulation for the admin includes course addition/deletion system and booked appointment deletion. The mobile application has a user-friendly interface with features such as secure login feature, social media integration, one to one booking system, an integrated geolocation functionality and dynamic data manipulation. Furthermore, the mobile application will only be compatible with iOS and will be available for download on the App Store. The project will also include testing and quality assurance to ensure the application is user-friendly, bug-free, and performs well on different devices.

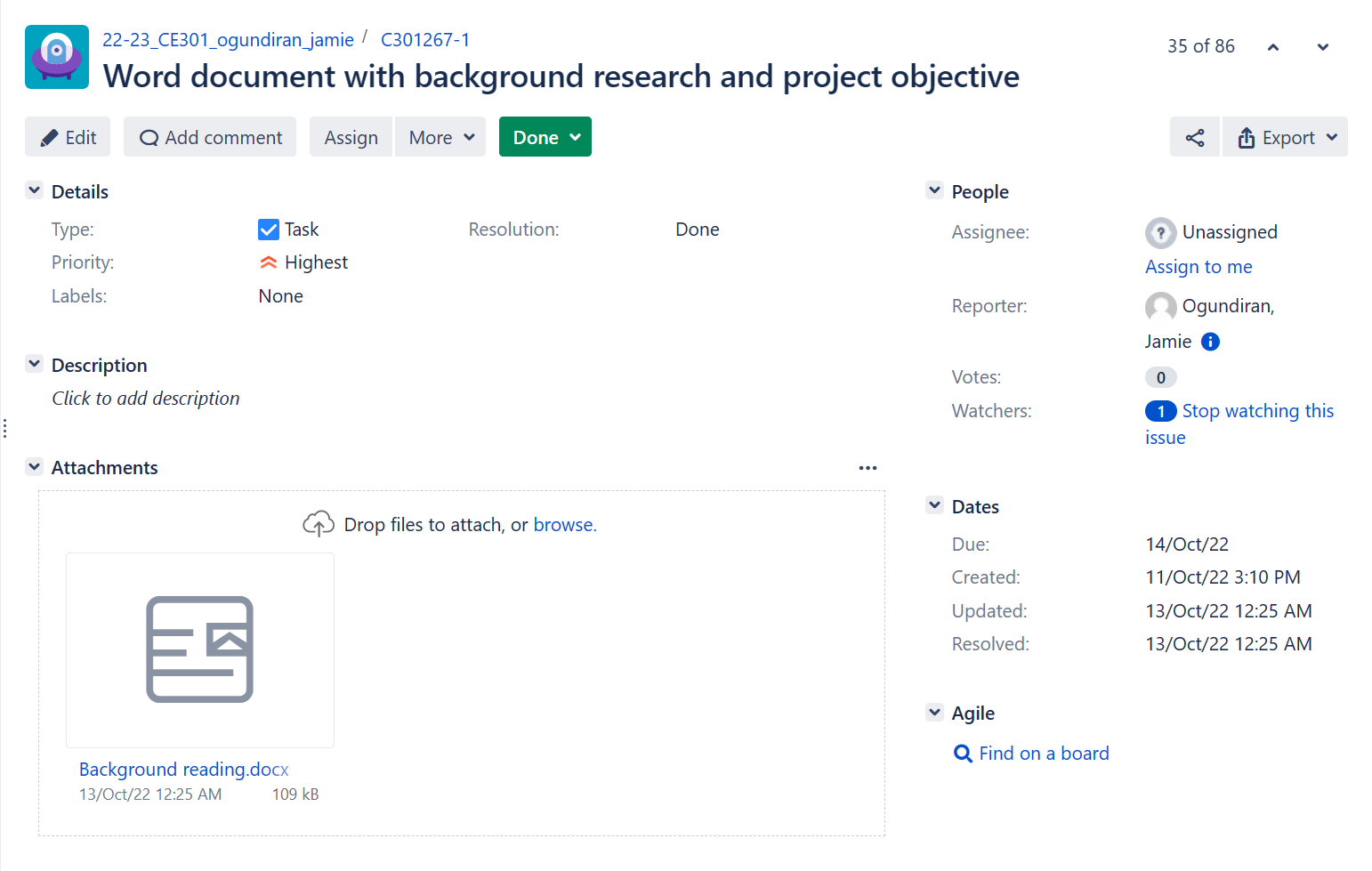
## 7.2 Project Management Methodology

The agile methodology was employed to direct the development of the mobile application. A Kanban board, managed through Jira software, was utilized to assign, and track project tasks, bug reports, and project stories. The reason behind adopting agile methodology as our project management approach is due to it benefits over the other methodologies. These benefits are iterative approach which is crucial in making improving the technical and user-experience aspect of the application by receiving feedbacks from the client and the supervisor, enhanced collaboration with the client which allows for a daily meeting with the client to make sure the client is satisfied with the product, early delivery of value which provides value to the client sooner, continuous improvement which emphasize regular adjustments to improve the optimization of the application, and better risk management which identifies any potential risks early on to reduce project failure.

Overall, the use of the agile methodology with a Kanban board managed through Jira software has proven to be the optimal methodology for the development of the mobile application. The benefits of using this approach include iterative approach, enhanced collaboration, early delivery, continuous improvement, and better risk management. These benefits combined make the agile methodology the optimal choice for project management.

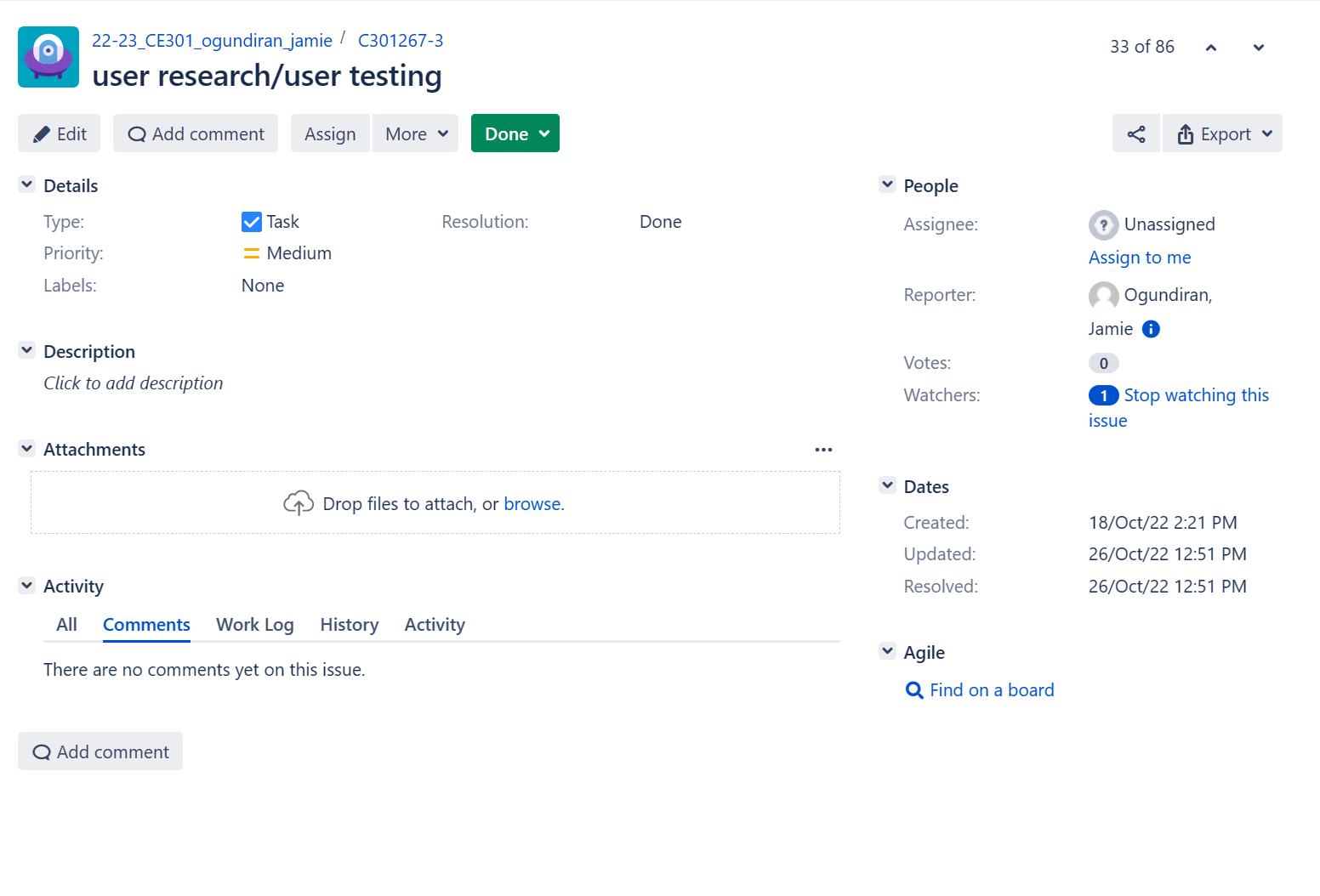
## 7.3 Project TimeLine (Gantt Chart)

A Gantt chart is a graphical representation of a project schedule that shows the timeline of a project's tasks, their dependencies, and the progress made on them. The Gantt chart above displays the timeline of tasks of my project. The task includes requirement analysis, user research, mobile app design, app development, evaluation and testing and project report. The requirement analysis which consists of gathering information, objective identification and documenting it. The requirement analysis started from October 03, 2022, to around October 20th, 2022.



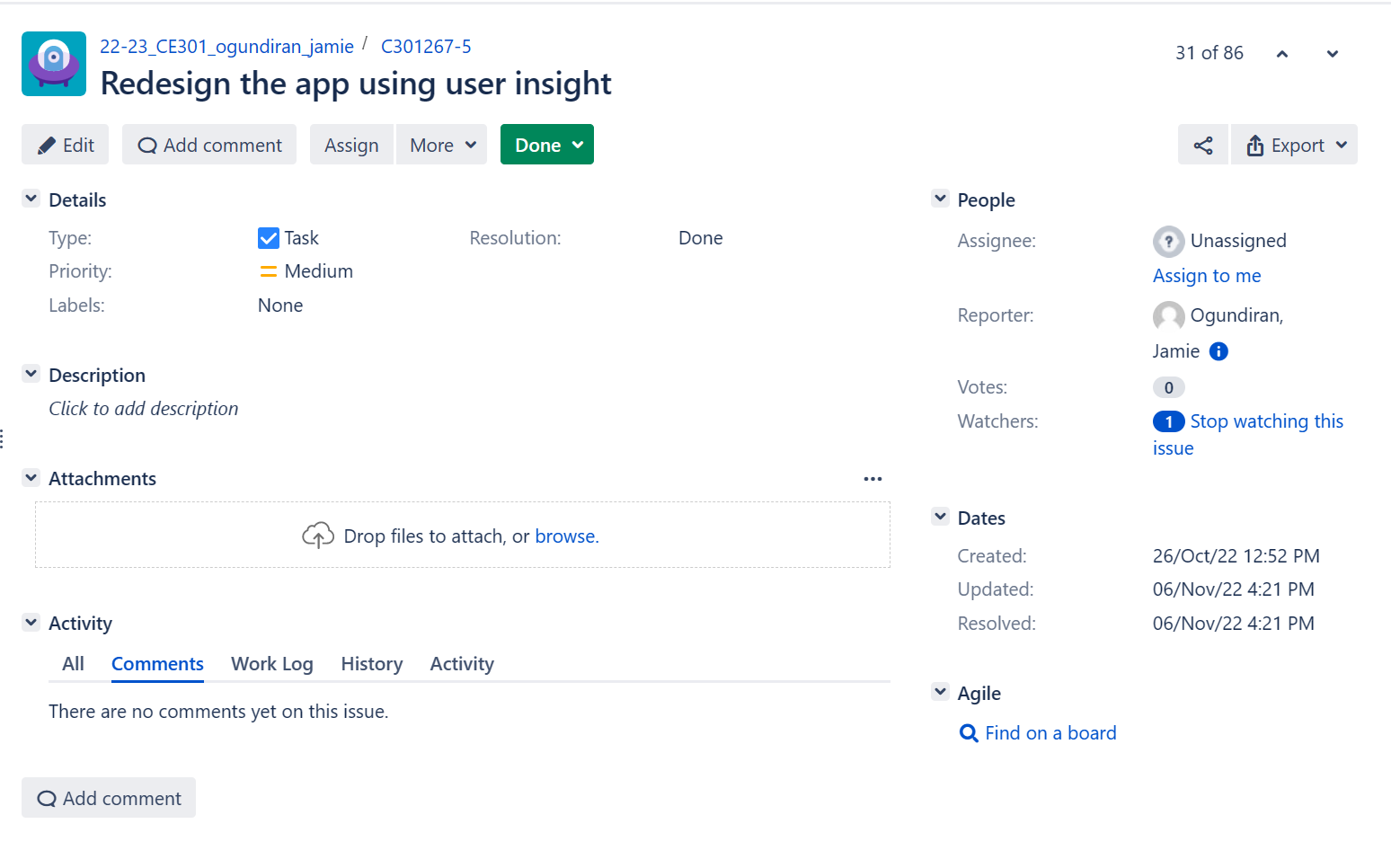
The task management is documented using JIRA like in the diagram above. The diagram above showcases the completion of requirement analysis, the research and project objective were written in word document and presented to the client via meeting.

The user research phase was conducted with utmost care and attention to detail, from approximately October 21st to October 30th.



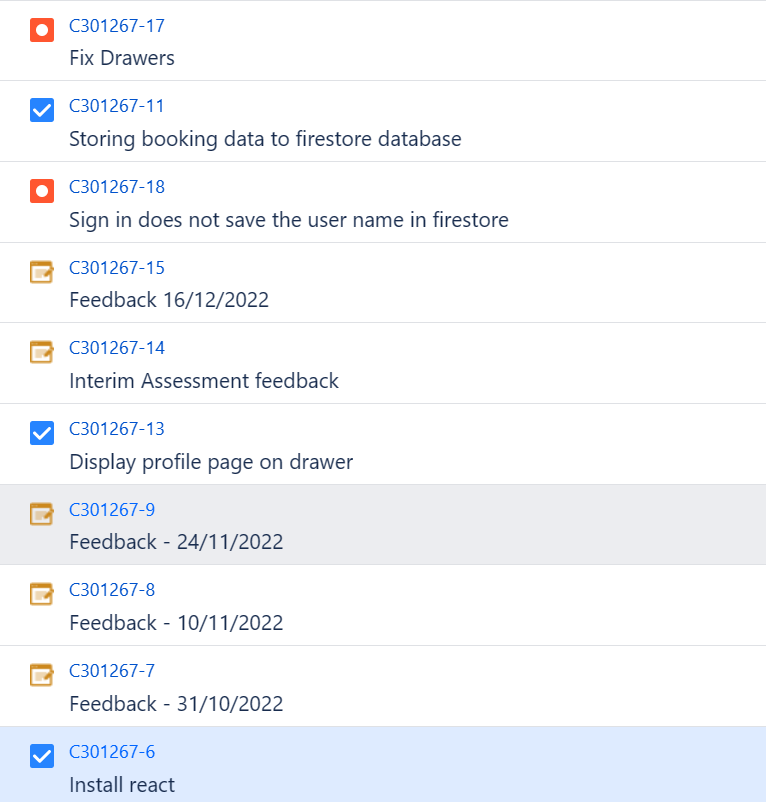
The diagram above showcases the completion of user research, the user research involved conducting comprehensive end-user interviews, analysing the competitive landscape, and creating detailed user use cases for thorough analysis.

The next phase involved crafting a top-notch app design, encompassing User Interface (UI) and User Experience (UX) design, as well as the creation of wireframes and mock-ups. This phase took place from November 1st to November 8th.



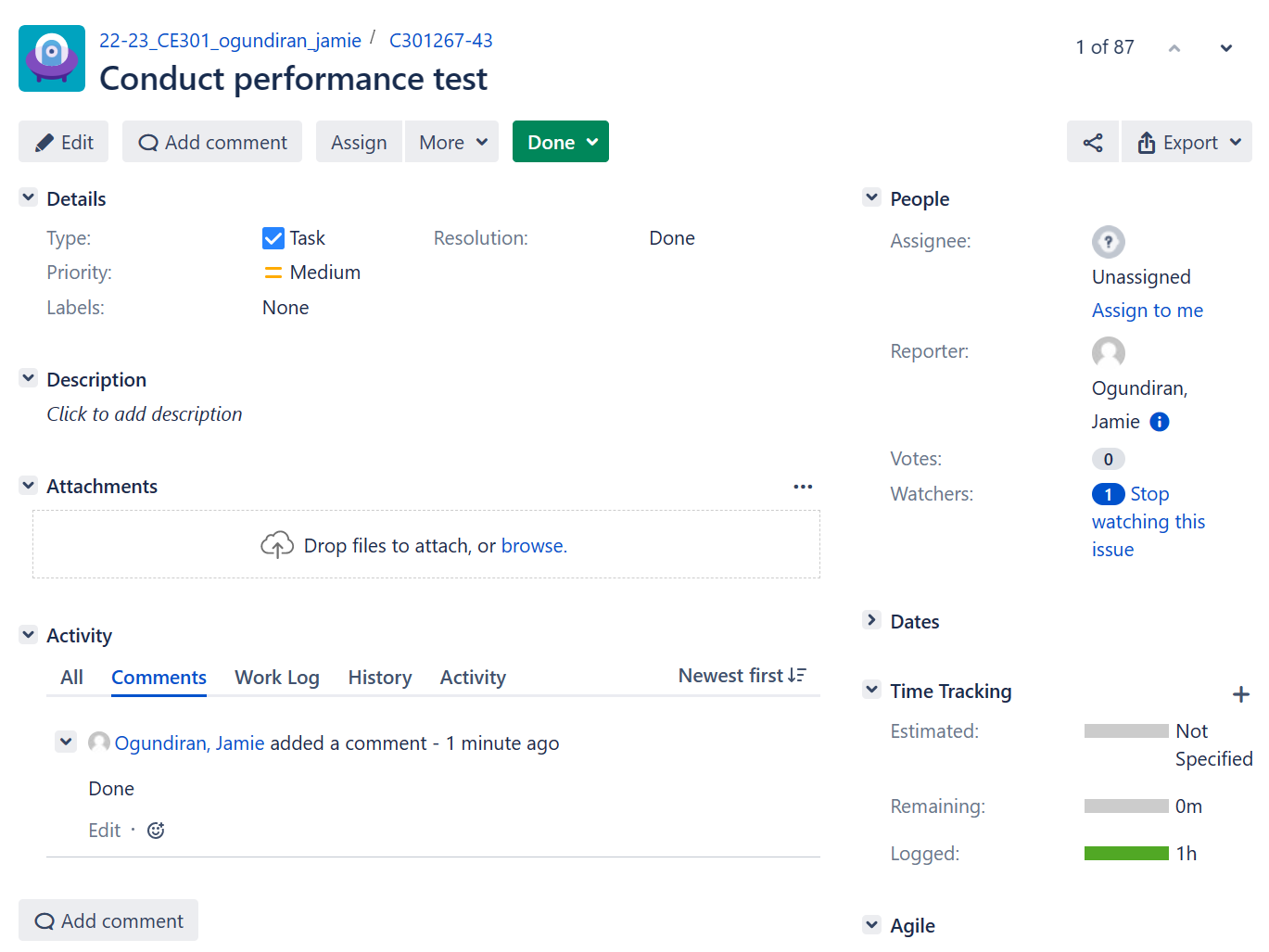
The design of the app was completed on November 06th, the design was completed with the help of the user insight obtained.

With the design in place, the app development phase kicked off on November 8th and continued until April 28th. This involved implementing cutting-edge programming software and bringing the app to life with technical finesse.



During the app development phase, several tasks were created to ensure that the project progressed smoothly and efficiently. While I cannot display all of them in the diagram above, the ones depicted highlight the breadth and depth of the work that was required. In addition, I utilized the Kanban board's bug tracking feature to highlight an issue that needed to be addressed promptly. This approach helped to ensure that the final product was of the highest quality, and that all stakeholders were satisfied with the result.

Throughout the development process, Evaluation & Testing was carried out in parallel with program testing and user testing to ensure the app met the highest standards of quality and user experience.



Performance test is one of the tests I conducted during the app development phase, The test conducted to find optimization issues and improve the app’s efficiency with the gathered information.

Finally, the project report was initiated around January, midway through the app development timeline, to provide a comprehensive overview of the project and its outcomes. The entire project was executed with a meticulous approach and unwavering commitment to delivering an exceptional app to users.

## 7.5 Project Risk Management

In mobile app development, managing risks is crucial for success. By identifying and addressing potential problems early on, you can increase your chances of meeting stakeholder expectations and avoiding project failure. Potential risks such as technical issues, management issues and release issues could jeopardize the success of your application. To avoid project failure, I used the minute strategy where I brainstormed to come up with a mitigation process for the risks identified. potential risks, poor UI/UX integration, Inadequate Testing, Delayed Release, Unreliable Data Storage, developing an application that is not valuable to the users and Stuffing Several Functions into a Single Programme.

Management Risk

* Risk: delays in the project due to a health issue

Mitigation ways: no ways of mitigation for the risk

Action plan: Work during vacation/break for the time lost for the delay.

* Risk: Changes in scope of the project.

Mitigation ways: Conduct a meeting with the client often to receive an approval of the changed scope. Notify the client about the changelog and how the change affected the roadmap.

Action plan: Review the importance of the changes, why it had to be changed? And implement it for the next sprint.

* Risk: Delays in feedback/approval from the client.

Mitigation ways: Communicate with the client regularly and inform the affect delay could causes.

Action plan: Remind the client during every meeting.

* Risk: Poor UI/UX integration.

Mitigation ways: conduct user research and go through several iteration of app design such as wireframe, prototype, and mock-ups.

Action plan: evaluate the user research conducted and solve the issue highlighted, redesign the app utilizing the information obtained from the user research.

Technical Risk

* Risk: Unreliable Data Storage

Mitigation ways: Research for the best performing data base management system (DBMS) for the mobile application.

Action plan: evaluate scalability or connectivity of the code that is causing the poor performance in data storage, and solve the issue

Risk: Platform compatibility

Mitigation way: Conduct thorough testing to ensure the app functions properly on all target platforms

Action plan: Evaluate the platform the application is not supported in and optimise it.

* Risk: Security concern

Mitigation way: Follow secure coding practices and incorporate encryption and other security measures to protect user data.

Action plan: Evaluate the method used to breach the security and adapt the code to improve and handles cyber-attacks.

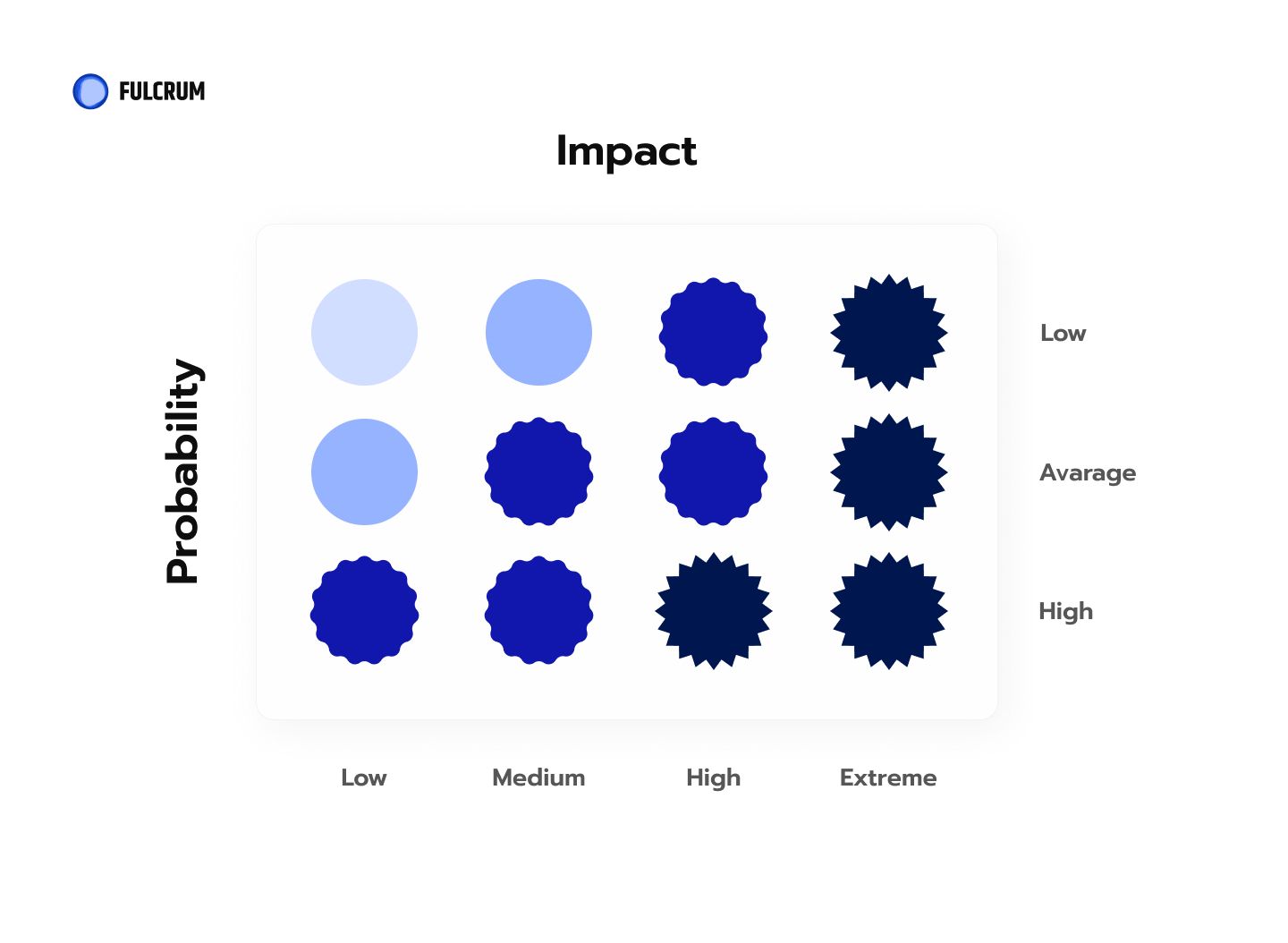
User Acceptance Risks

* Risk: Negative user feedback on app’s features, usability, and performance

Mitigation ways: conduct adequate amount of user tests on the features usability and the performance of the app

Action plan: Answer user feedback, fix problem, reward a person, who found an issue (if a problem is in some issue).

### 7.6 Risk Evaluation



The diagram above displays the level of probability and impact a risk. The prioritisation and ranking of a risk are created using the information from the diagram above. The prioritising of a risk becomes higher as the impact of the risk is more and the probability of occurrence is high.

|  |  |  |  |
| --- | --- | --- | --- |
| RISK | IMPACT | PROBABILITY | Overall Risk Rating |
| Management risk |  |  |  |
| Delay in project due to health issue | Low | Low | Low |
| Changes in scope of the project. | High | Average | High |
| Delays in feedback/approval from the client. | Low | Average | Medium |
| Poor UI/UX integration | High | High | Extreme |
| Technical risk |  |  |  |
| Unreliable Data Storage | Extreme | High | Extreme |
| Platform compatibility | Extreme | Average | High |
| Security concern | High | Average | High |
| User Acceptance Risks |  |  |  |
| Negative user feedback on app’s features, usability, and performance | High | High | Extreme |

The table above displays the risks and their risk rating, The risks that needs to be maintained the most are poor UI/UX integration, unreliable data storage, negative user feedback on app’s features and usability and performance. The risks that are on the second priority of maintenance are platform compatibility and security concern. Moreover, the risk that is on the third priority of maintenance is delays in feedback/approval from the client and lastly the risk that has the least priority on maintenance is delay in project due to health issue.

# 8.Evaluation & Testing

Evaluation is a crucial step in mobile application development for the application to succeed upon launch. Evaluation consists of code testing which includes bug checks and performance testing, user testing which include testing application usability. The code testing was done using the Jest testing framework and react native renderer. And the user test was done with questionnaires and interviews.

### 8.1 Unit Test

A unit test is a type of software testing that involves testing individual units or components of a software system in isolation. The purpose of a unit test is to verify that a particular unit of code, such as a function or method, is working as intended and producing the expected output given certain inputs. We created a unit test for every function available on our code.

### 8.2 Performance Test

Expo which is set of tools and services that simplify the process of building, deploying, and testing React Native apps provides a built-in performance monitor feature that allows the developers to monitor app’s performance in real-time. In this section I will be utilising the feature to document the CPU and memory usage in my application. The measurement is scaled by frames per second (FPS).

#### I/O test

To assess the performance of the app's I/O, I conducted a series of tests using the text inputs on the login and sign-up pages. Each test was conducted 10 times for both the email and password text inputs, and the average frames per second (FPS) was recorded for analysis.

The results showed that the app's user Interface (UI) had an average FPS of 52, while the Java Script (JS) had an average FPS of 55. Additionally, the FPS quickly returned to 60 after each input, indicating that the app's I/O performance is exceptional.

These findings suggest that the app's text input functionality is optimized for fast and efficient performance, with an average FPS of 55 and 52. Overall, the results are a testament to the quality of the app's design and development and provide valuable insights for further optimization and improvement.

#### Navigation

Similarly, to the test I conducted on I/O test, I conducted a series of tests navigating between the screens on the app. The navigation between tab navigator, drawer navigator and stack navigator were tested on this test.

The test results revealed that the tab navigator's user interface (UI) had an average frames per second (FPS) of 57, while the Java Script (JS) had an average FPS of 56. Similarly, the drawer navigator's UI had an average FPS of 55, while the JS had an average FPS of 57.

On the other hand, the stack navigator's UI had an average FPS of 58, while the JS had an average FPS of 59. These findings suggest that the app's navigators are optimized for fast and efficient performance, with the stack navigator outperforming the other navigators in terms of FPS.

#### Flat list

When testing the performance in flat list, I recorded the FPS of the UI and JS by refreshing the app and re-rendering the flat list each test. The flat list in booking page was used to conduct the test.

The results of the tests showed that the flat list's UI had an average frames per second (FPS) of 59, while the JS had an average FPS of 54. This indicates that the app's flat list performs exceptionally well in terms of UI but experiences a slight drop in FPS in the JS.

#### Buttons

When testing the performance of the button component in my app, I recorded the FPS of the UI and JS by toggling between” add” and “added” of the button in the first booking page and recording the FPS each toggle.

The results of the tests showed that the button component had an average FPS of 55, while the JS had an average oof 49. We can conclude from the results that the app’s button component performs well however there is a little bit of a spike in the FPS of JS when switch between toggles.

### 8.3 User Test

User testing is a critical component of mobile app development, as it can have a significant impact on the success of an app. The mobile app market is highly competitive, and users expect apps that are intuitive, user-friendly, and provide a seamless experience. A lack of user testing could result in usability issues and poor user experience, leading to negative feedback and low user retention. To ensure optimal performance and a positive user experience, it is crucial to conduct user testing during the app development process. By doing so, developers can gain valuable insights into user behaviour, preferences, and identify areas for improvement. I conducted a user testing exercise on my mobile app using a questionnaire, designed to test usability and user experience.

The questionnaire we used for this experiment is a questionnaire focused on the user experience called UEQ (user experience questionnaire), since the app is targeted toward English speaker, the questionnaire will be provided in English. The questionnaire uses a sematic differential as the measurement scale and its goal is a fast direct measurement of UX.



To achieve more accurate and detailed result, I conducted an interview including 6 questions concerning the usability of the app. The questions were provided from playbox UX. (UX, n.d.)

Here are the questions asked:

1. What parts of the web/mobile app did you like the most? Why?
2. What parts of the web/mobile app did you use the least? Why?
3. What did you think of the interface?
4. What do you think about the way features and information were presented?
5. Why will you keep using this web/mobile app? Why will you not?
6. How did you first learn about our product?
7. How likely are you to recommend our product to a friend or family member?

#### Test Result

The test includes a questionnaire, I tested 5 possible customers and 3 possible employees to test the usability of the app.

Using the UEQ benchmark system created by Martin Schrepp, Andreas Hinderks and Jörg Thomaschewski (Martin Schrepp, Andreas Hinderks and , Jörg Thomaschewski, 2017), I calculated the overall user experience score for the user experience questionnaire (UEQ). The UEQ benchmark system is a useful tool for evaluating user experience as it provides a standardized way to compare your results with a large dataset of other studies.

Based on the UEQ benchmark system, the results fall into one of five categories: excellent, good, average, poor, or bad. An overall score of 1.00 is considered excellent, 0.80 is good, 0.60 is average, 0.40 is poor, and 0.20 is bad.

Moreover, the questionnaire with 26 items is grouped into six scales, Attractiveness, Perspicuity, Efficiency, Dependability, Simulation and Novelty. The score is calculated by summing up the scores for the relevant items and then normalizing it to a scale of 0.20 to 1.00.

Here is the result I obtained from utilising the UEQ benchmark system.

# Conclusion

# Bibliography

alvin. (2021, 04 19). *How Many Apps Are There Globally? (2021 Facts and Statistics)*. Retrieved from MobileApps: https://www.mobileapps.com/blog/how-many-apps-are-there

Gharajedaghi, J. (2011). Systems Thinking, Third Edition: Managing Chaos and Complexity: A Platform for Designing Business Architecture. In J. Gharajedaghi. Morgan Kaufmann Publishers.

Margaretha Ohyver, Jurike V. Moniaga, Iwa Sungkawa, Bonifasius Edwin Subagyo, Ian Argus Chandra. (2019). The Comparison Firebase Realtime Database and MySQL Database Performance using Wilcoxon Signed-Rank Test. (p. 396). Indonesia: Elseview B.V.

O'Dea, S. (2022, November 17). *Smartphones-Statistcs and Facts*. Retrieved from statista: https://www.statista.com/topics/840/smartphones/#topicHeader\_\_wrapper

Soininen, V. (2021). Jetpack Compose vs React Native – Differences in UI Development. *Metropolia University of Applied Science* (p. 32). Metropolia University of Applied Science.

Wu, W. (2018). React Native vs Flutter, cross-platform mobile . (p. 28). Metropolia University of Applied Sciences.