VIETNAM NATIONAL UNIVERSITY OF HOCHIMINH CITY
THE INTERNATIONAL UNIVERSITY

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

MOBILE TELEHEALTH SYSTEM (MOBIHEAL)

By
Enh Uong Family

A report submitted to Dr. Le Duy Tan - Lecturer of the School of Computer Science and Engineering
in partial fulfillment of the requirements for the assignment of Mobile Application Development course in Semester 1 (2023 - 2024)

Ho Chi Minh City, Vietnam
2024

MOBILE TELEHEALTH SYSTEM
(MOBIHEAL)

APPROVED BY:

_____,
Le Duy Tan, Dr., Lecturer

_____ ,
Tran Ton Dai Nghia, Leader, ITITIU20148

_____,
Dang Nhat Huy, Member, ITITIU20043

_____,
Bui Thi Cam Van, Member, ITITIU20111

_____,
Mai Dang Huy, Member, ITITIU20046

_____,
Huynh Tan Thien, Member, ITITIU20020

_____,
Nguyen Huynh Nguyen, Member, ITITIU20261

REPORT COMMITTEE

ACKNOWLEDGMENTS

We would like to express our heartfelt thanks to Dr. Le Duy Tan for allowing us to participate in this project and enabling us to translate our theoretical knowledge into practical applications. Although challenging, this project has proven to be extremely rewarding.

Our appreciation also extends to the faculty of the School of Computer Science at the International University, with a special mention to Dr. Le Duy Dan, who has supported us from the start of our academic journey.

Mr. Tan, we are grateful for your dedication and expertise in teaching. We are committed to ensuring that the knowledge we have gained is effectively utilized and continuously improved upon.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

In response to the role of technology in healthcare, the "Mobile Telehealth System" (MOBIHEAL) was meticulously developed using Android Studio, Kotlin, and Firebase. This innovative application, designed to revolutionize telemedicine services, integrates advanced features such as appointment booking, sophisticated sorting algorithms, digital prescription management, health statistics visualization, UPI payment integration, and a dynamic patient wait list. The development methodology employed a comprehensive approach to address distinct telehealth needs, ensuring the creation of a user-friendly, secure, and efficient platform.

The development of MOBIHEAL's features was driven by a strategic approach, with each element crafted to meet specific telehealth requirements. The appointment booking system, supported by Firebase's real-time database, provides a seamless user experience while prioritizing data security. A unique sorting algorithm enhances healthcare efficiency by prioritizing appointments based on various health parameters. The system's digital prescription management ensures easy access to medical records, incorporating robust security measures. Health statistics visualization empowers users with insights into their health trends, fostering a proactive approach to personal well-being. The integration of instant UPI payments expedites financial transactions, and the dynamic patient queue list leverages Firebase's capabilities for real-time updates, optimizing scheduling efficiency.

MOBIHEAL's user requirement analysis reflects a thoughtful and user-centric design. For patients, the application offers a user-friendly interface for appointment scheduling, prioritized waitlists based on severity, secure digital prescription management, and personalized health data monitoring. The emphasis on security protocols ensures the confidentiality of patient information. In parallel, the system caters to the needs of healthcare professionals by providing an effective appointment scheduling tool, real-time patient queue management, secure access to patient health data and prescriptions, and a robust feedback system. By fulfilling these user requirements, MOBIHEAL aims to redefine healthcare administration, offering a seamless and convenient experience for both patients and healthcare professionals.

CHAPTER 1

INTRODUCTION

1.1. Background

Technology integration is becoming more and more important in the quickly changing healthcare sector, particularly in telehealth services. The emergence of mobile health applications like Mobiheal represents a paradigm change toward more easily accessible, effective, and patient-focused healthcare. Mobiheal is supported by Firebase and is written with Kotlin in Android Studio. Appointment scheduling, appointment sorting algorithms, digital prescription management, health data visualization, rapid UPI payment integration, and dynamic patient queue management are just a few of the features it has. These elements are intended to improve patient and physician experiences with healthcare delivery and expedite telemedicine services. (TIGCAL)

1.2. Problem Statement

Although telehealth services have great potential, several obstacles prevent their widespread adoption and efficacy. These include the intricacy of maintaining and making appointments, the challenge of ranking urgent cases, the ineffective management of medical records, and the deficiency of real-time patient-provider contact. (Guest) Furthermore, the traditional payment and prescription administration procedures are frequently complicated and unintuitive. By offering a complete solution that makes use of technology to streamline and enhance the telehealth experience, Mobiheal seeks to address these issues. (Varaksina)

1.3. Scope And Objectives

The creation and implementation of the Mobiheal application are included in the project's scope, with a particular emphasis on:

- *Simplifying Appointment Scheduling*: Making it simple for patients to locate and make medical provider appointments.
- *Optimizing Appointment Sorting*: Putting in place an algorithm to rank appointments according to health-related criteria, guaranteeing that urgent situations are attended to promptly.
- *Digital Prescription Management*: Providing safe digital prescription posting and retrieval for convenience and better healthcare delivery.
- *Health Statistics Management*: The visualization of health statistics can help individuals manage their health by giving them the means to enter and track their health data.
- *Seamless Payment Integration*: Integrating a UPI payment system for simple and rapid transactions is known as seamless payment integration.

- *Effective Queue Management*: Using a dynamic patient queue system can improve scheduling efficiency for physicians and patients alike. This is an example of effective queue management.

The principal aims of this initiative are to optimize telemedicine service efficiency, augment patient and provider satisfaction, and guarantee system scalability and security.

1.4. Assumption And Solution

- Assumption:
    - *User Compliance*: It is presumed that users-physicians and patients alike-will be amenable to embracing and adjusting to a novel digital healthcare service.
    - *Technological Accessibility*: To use the program efficiently, users must have access to Android devices and internet connectivity.
    - *Regulatory Compliance*: The program complies with all applicable data protection and healthcare requirements.
- Solution:
    - *User-Friendly Design*: The simple and easy-to-use interface of Mobiheal promotes adoption and lowers the learning curve that comes with utilizing new technology.
    - *Strong Backend Support*: The application is scalable, safe, and effective when using Firebase for real-time data management.
    - *Continuous Improvement*: Using evolutionary prototyping in conjunction with an agile development approach enables iterative feedback and ongoing application enhancement to successfully satisfy user needs.
    - Protecting sensitive health data and user information by putting advanced security processes and compliance mechanisms in place.
    -

Mobiheal wants to raise the bar for telehealth services by providing strong answers to these presumptions, improving the accessibility, effectiveness, and user-friendliness of healthcare.

CHAPTER 2

LITERATURE REVIEW/RELATED WORK

2.1. Review 1

In our pursuit to develop an exemplary mobile health system, we chose to appropriate *ContinuousCare* for its robust telehealth features and virtual care capabilities. Its comprehensive approach to patient engagement and health management through a digital platform serves as an ideal reference for integrating remote healthcare services into our app.

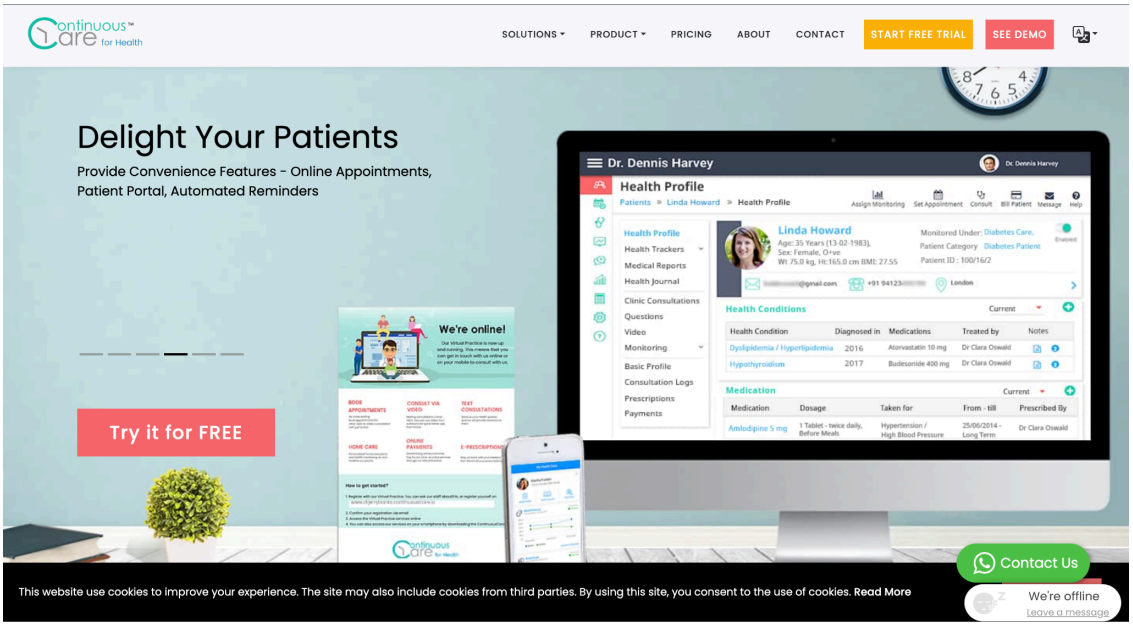

Figure 2.1. ContinuousCare Homepage

ContinuousCare presents a comprehensive digital health platform tailored for clinics and businesses, positioning itself as the quintessential software needed for modern healthcare management. With a suite of services that includes book appointments, video consultations, text consultations, home care, online payments, and e-prescription, *ContinuousCare* offers a multifaceted approach to healthcare. The platform notably addresses the increasing demand for virtual healthcare services, providing an array of tools that cater to both patient convenience and clinical efficiency. (Varma)

2.1.1. Sub Review 1

ContinuousCare offers an array of features designed to facilitate comprehensive clinic management and telehealth services:

- *Virtual Practice*: Online and mobile platform for health providers to engage with their patients.
- *Appointment Booking*: Automated notifications and reminders for both patients and providers.
- *Video Consultations*: Secure video consultations for remote patient engagement.

- *Text Consultations*: The platform supports text consultations for ease of communication.
- *Remote Monitoring*: Health tracking and chronic care management for various conditions.
- *Patient Health Records*: Integrated management of patient records.
- *Billing and Payments*: In-app billing and payment processing.
- *Prescriptions and Lab Order Support*: Simplifies the process of managing prescriptions and lab orders.
- *Patient Engagement Features*: A suite of features to enhance patient engagement with healthcare providers.

Figure 2.2. ContinuousCare - Personalized Business UI

Delving into the user experience, the *ContinuousCare* system showcases an intuitive dashboard that centralizes patient information, including health profiles, medical reports, and health journals. For instance, the platform facilitates a seamless user journey from booking an appointment to consulting with healthcare professionals via video. Moreover, this integration extends to medication management, where digital prescriptions are easily accessible and manageable within the platform. To interpret, the user interface, as depicted in the image, indicates a clear, structured layout with a focus on ease of navigation, which is likely to reduce barriers to technology adoption among patients and healthcare providers. (Varma)

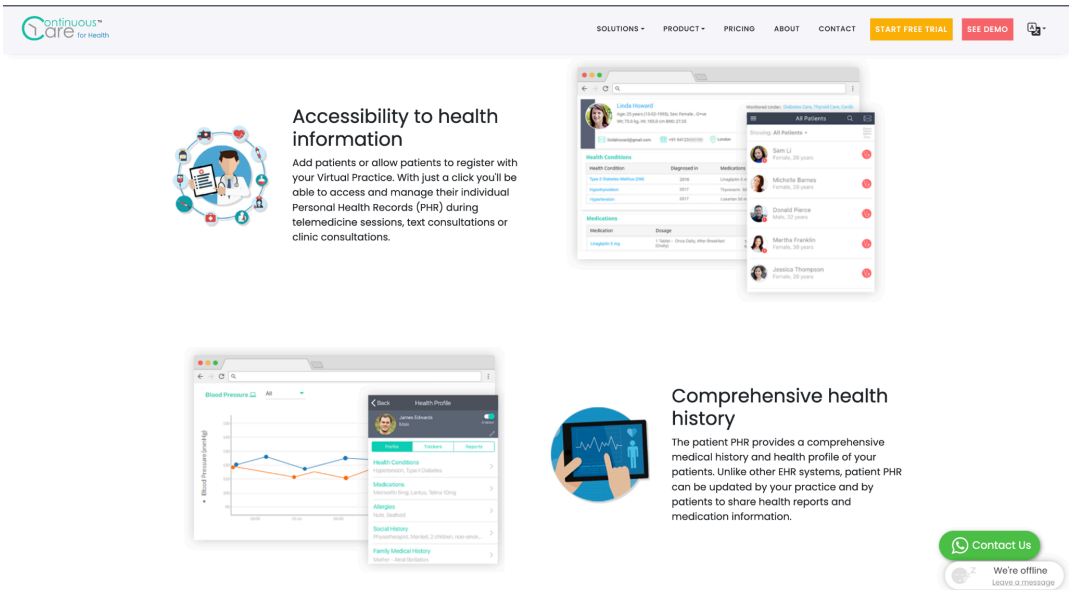

Figure 2.3. ContinuousCare - Patient Personal Health Record

In this subsection, you could further analyze the integration of various functionalities, such as how the e-prescription system interfaces with appointment scheduling and payment systems, providing a seamless end-to-end experience for managing a patient's care journey. ContinuousCare has a robust system for ensuring that patients can manage their health profiles and engage with healthcare providers efficiently, which is critical in the context of mobile health systems. (Varma)

2.2. Review 2

Conversely, *Tam Anh Hospital's Health System* provides a contrasting case study with its advanced in-hospital technology integration and luxury amenities. Analyzing this system will help us understand how to incorporate high-end, patient-centric features into our mobile health system, ensuring a seamless in-person experience.

Figure 2.4. Tam Anh's Homepage

Tam Anh Hospital showcases a holistic healthcare approach through its digital presence, emphasizing comprehensive child care. The hospital's online platform facilitates appointment booking with a selection of locations and services, reflecting a patient-centric model that values accessibility and convenience. With the online platform, the system offers a range of specialized services, underpinned by a team of experienced medical professionals and a modern, technologically equipped infrastructure. Furthermore, This platform likely serves as a bridge, reducing the gap between patients and healthcare services by streamlining the appointment process through digital means. (Hospital)

2.2.1. Sub Review 2

Tam Anh Hospital provides a health system with modern equipment and specialized services that cater to patient needs and efficient healthcare delivery:

- *Electronic Identification Numbers*: Assigns electronic ID numbers for patient record management across all hospital activities.
- *Streamlined Procedures*: Simple and closed-loop procedures at dedicated floors to minimize patient time and provide same-day examination and test results.
- *Specialized Department Coordination*: Close coordination among various specialized departments such as IVF, orthopedics, and emergency care to optimize treatment effectiveness.
- Customer Care Services: 24/7 customer care support via hotline, website, page, and smart applications.

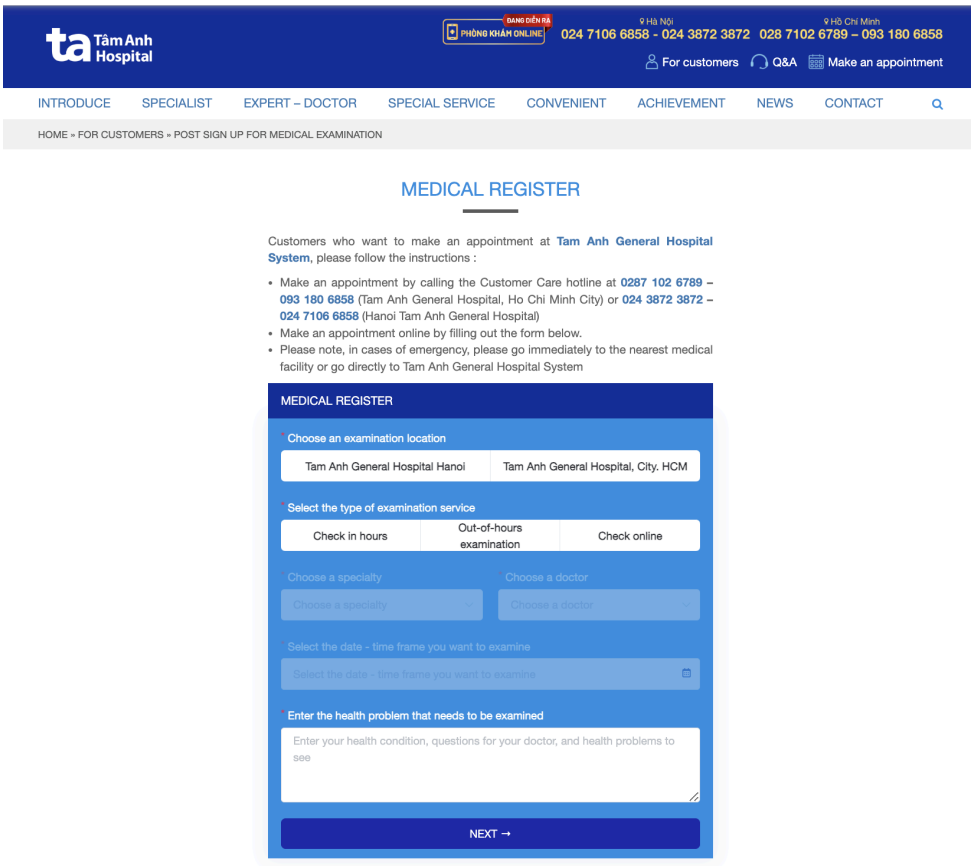

Figure 2.5. Tam Anh - Session Schedule Form

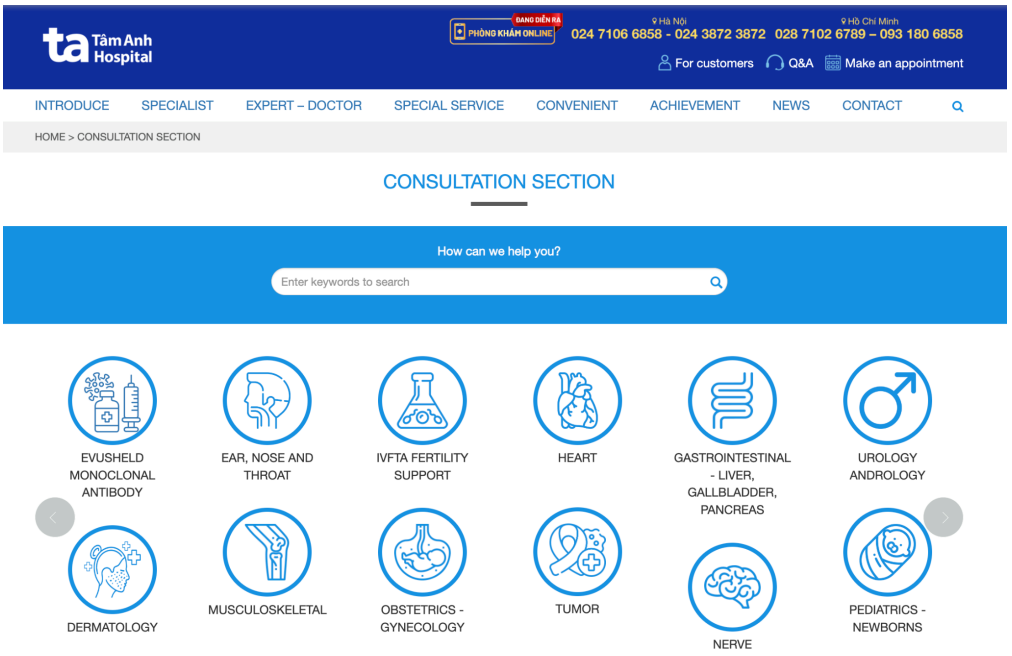

Figure 2.6. Tam Anh - Fields Online Consultant Service

The user interface of *Tam Anh Hospital's website* provides insights into the effectiveness of their patient engagement strategies. It features prominent calls to action, such as direct appointment booking options and easily accessible online consultation services. Plus, the hospital's commitment to customer care is evident through dedicated customer service channels, including a chat function, suggesting an interactive and responsive patient support system. This engagement is likely to enhance the patient experience by providing timely assistance and personalized care pathways. Moreover, the visible inclusion of distinguished medical personnel on the platform could instill trust and assurance in the quality of care provided. (Hospital)

Furthermore, the hospital's patient engagement metrics and feedback mechanisms would be beneficial to understand the impact of these digital tools on patient satisfaction and health outcomes.

To conclude, both platforms offer features that enhance the patient care experience through the use of modern technology. *ContinuousCare* focuses on virtual engagement and management of healthcare services, while *Tam Anh Hospital* emphasizes both technological advancements and online amenities for patient care. Each system addresses different aspects of patient care, with *ContinuousCare* leaning towards telehealth and remote monitoring, and *Tam Anh Hospital* providing a comprehensive in-hospital experience backed by advanced security and customer care services. These features reflect the diverse approaches to integrating digital solutions in healthcare to improve patient outcomes and service quality. That is why we sample the mentioned 2 platforms to implement into our app *MobiHeal.*

CHAPTER 3

METHODOLOGY

3.1. Overview

Technology integration into healthcare services is becoming critical in the quickly developing field of telehealth. This connection created motivation for us to create the "Mobile Telehealth System" (Mobiheal), an application created in Android Studio using Kotlin and backed by Firebase. The program seeks to simplify telemedicine services with its array of features, which include appointment booking, sorting algorithms, digital prescription accessibility, health statistics, fast UPI payment integration, and an effective patient wait list. This study explores the development processes used to create this novel system.

Mobiheal's features were all painstakingly created to meet distinct telehealth needs:

1. *Appointment Booking*: Patients can look up doctors and make appointments using the user-friendly interface that was created for them. Users' important health information was gathered using a form-based technique and safely stored in Firebase's real-time database.
2. *Appointment Sorting Algorithm*: A unique algorithm was created to analyze user data and prioritize appointments based on various health parameters. This algorithm ensured that critical cases received timely attention, thereby enhancing the efficiency of healthcare delivery.
3. *Digital Prescription Management*: Recognizing the importance of easy access to medical records, the system allowed patients to upload and store their prescriptions. These documents were encrypted and securely stored, with provisions made for doctors to access them during consultations.
4. *Health Statistics Visualization*: Users were given the ability to input and track their health data, which was then visualized using advanced charting techniques. The purpose of this function was to give consumers insights into their health trends.
5. *Instant UPI Payment Integration*: To expedite the payment process, an instant UPI payment system was incorporated. This function automatically generated QR codes that were connected to doctors' UPI IDs, therefore reducing the pain that comes with financial transactions in hospital settings.
6. *Dynamic Patient Queue List*: Leveraging Firebase's capabilities, the application provided real-time updates on patient queues, enhancing the scheduling efficiency for doctors and patients.

The development of Mobiheal, a telemedicine application, adhered to strict privacy regulations and ethical principles, ensuring user consent and advanced security techniques. Utilizing Android Studio, Firebase, and Kotlin, the project set the standard for future advancements in digital healthcare services.

3.2. User Requirement Analysis

Mobiheal was designed with the requirements of healthcare professionals doctors and patients in mind. Through the management of medicines, appointment scheduling, patient queue prioritization, and health data monitoring, the program seeks to transform the conventional approach to healthcare administration. The main features and specifications are described in this user requirement analysis, taking into account the views of both users.

3.2.1. Patient User Requirements

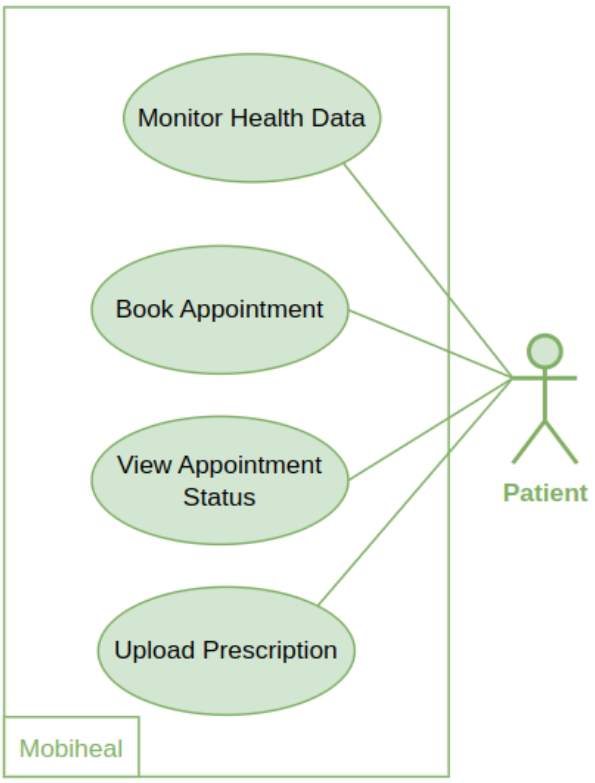

Figure 3.1. Patient User Goal Use Cases

Mobiheal's functionality and design are painstakingly customized to match each patient's unique requirements, guaranteeing a smooth and convenient experience when obtaining medical treatments (Figure 3.1). The ability for patients to easily schedule appointments with licensed doctors via the app is essential to meeting these standards. To enable patients to quickly search for and pick doctors, select available appointment slots, and receive prompt confirmation and reminders for their appointments, this function requires a simple and user-friendly interface *(Book Appointment)*.

It is also noteworthy how creatively the waiting line is managed. As opposed to first-come, first-served scheduling, Mobiheal gives patients priority based on the severity of their conditions. By setting priorities, all patients receive equal attention and have their requirements effectively met. To be successful, this system has to be open about how priorities are decided upon and give patients real-time information about where they are in the line and how long they should expect to wait. This degree of communication is essential for

controlling patient expectations and lowering waiting-related stress *(View Appointment Status)*.

Prescription management is another critical aspect of the patient user requirements. Mobiheal allows for the digital upload of prescriptions, with the system designed to retain only the most recent upload. By guaranteeing that the most recent medical information is easily accessible at the point of service, this feature streamlines the procedure for both doctors and patients. Strong security and confidentiality protocols are necessary to safeguard private health information associated with digital prescriptions *(Upload Prescription)*.

Moreover, the application offers a health data monitoring tool, enabling patients to input and keep track of their health metrics. The key to this feature is its visualization component, where patients can view their health data in the form of easy-to-understand charts, focusing on the latest five entries. This not only helps in self-monitoring their health conditions but also provides valuable data points during medical consultations. The ability to customize this feature according to different health parameters further enhances its usefulness, making it a versatile tool for personal health management *(Monitor Health Data)*.

The patient user requirements for Mobiheal reflect a thoughtful and user-centric approach, emphasizing ease of use, transparency, security, and personal health management. By fulfilling these requirements, the app stands to significantly improve the patient experience in accessing and managing healthcare services.

3.2.2. Doctor User Requirements

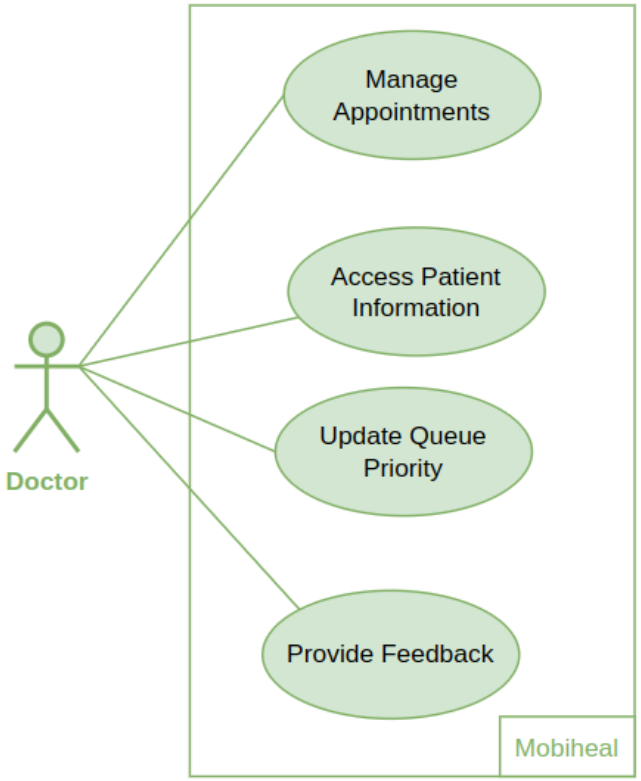

Figure 3.2. Doctor User Goal Use Cases

Mobiheal is intended to be a complete solution for doctors that simplifies many patient care and engagement tasks (Figure 3.2). An effective appointment scheduling tool is essential for doctors to use Mobiheal. With the use of this system, doctors should be able to see, manage, and modify their forthcoming appointments from a simple, well-organized interface. The user interface should be simple to use so that doctors can quickly go through their daily calendars and adjust their availability as needed. Furthermore, this function needs to provide the option to view patients' uploaded prescriptions and medical histories beforehand, enabling better planning and more efficient consultations *(Manage Appointments)*.

A sophisticated method for managing patient wait times is also essential. Doctors require an accessible and clear picture of the prioritized patient queue because of the app's novel approach to patient prioritization based on urgency rather than chronological order. In addition to showing the line, this system needs to let doctors change the order of priority as needed, particularly in emergencies or other urgent situations. For clinicians to effectively manage their time and attention, real-time updates on changes in the patient queue are crucial *(Update Queue Priority)*.

Prescription information and patient health data accessibility are also critical. doctors need an easy-to-use, safe way to access their patients' most recent medical records and medications. To maintain confidence and secrecy, this access must be instantaneous and by healthcare privacy regulations. To improve patient progress and treatment plans, the system should also enable doctors to comment or add notes to patient health data. The app should have a function that allows doctors to ask patients directly for more health information when it's necessary *(Access Patient Information)*.

Additionally, a feedback and reporting system is necessary for the app to be continuously improved and adjusted. The development team should be able to get input from doctors on the usability and functionality of the app, allowing for any necessary improvements. doctors would greatly benefit from having access to analytics and reports on appointment trends, patient demographics, and health data. By using this data, services may be better tailored to the requirements of the patient *(Provide Feedback)*.

The features that doctors need from Mobiheal include safe access to health data, patient and appointment scheduling, feedback systems, and more. These characteristics are essential to establishing a productive, safe, and easy-to-use environment for doctors, hence augmenting their capacity to deliver superior healthcare services via the application.

3.3. System Design

3.3.1. Architecture

The architecture of the Mobiheal developed with a focus on robustness and efficiency, significantly leverages the principles outlined in "Mastering Design Patterns" (Figure 3.3).

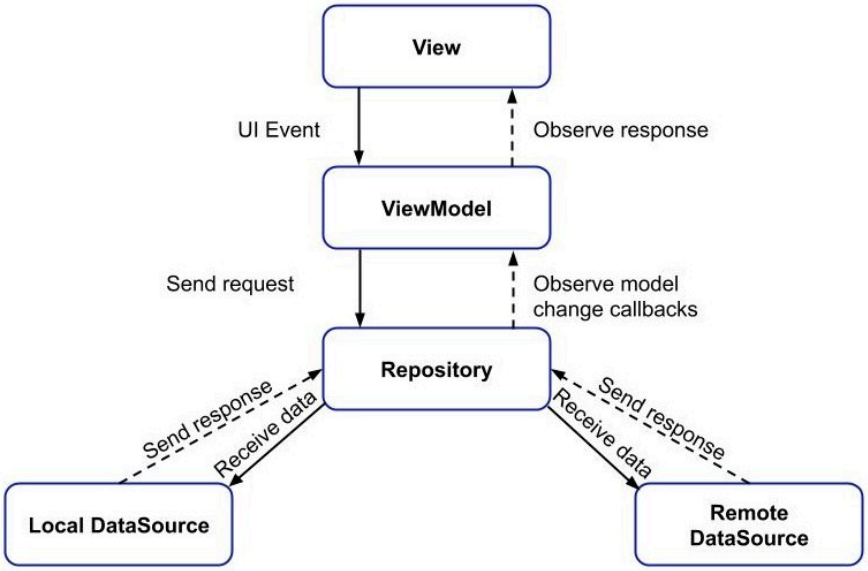

Figure 3.3. Mastering Design Pattern

At its core, the architecture divides the system into two main components: the frontend client-side and the backend server-side.

The front end was built primarily for Android devices using Kotlin and adopted various design patterns for optimal UI/UX. These include the Model-View-ViewModel (MVVM) pattern (Figure 3.4), ensuring a clean separation of the app's logic and the user interface. This separation enhances the app's maintainability and testability. We used the Retrofit library to integrate for efficient network requests and image loading, while local data caching and offline support are managed using the Room database, adhering to the Repository design pattern for data handling.

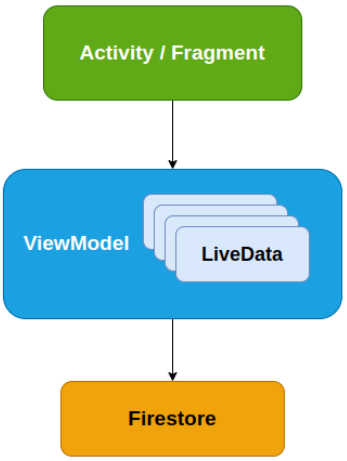

Figure 3.4. Android MVVM Pattern

The backend infrastructure is powered by Firebase, utilizing its suite of services to handle different aspects of the app. Firestore, a NoSQL database, is employed for real-time data storage and synchronization, fitting into the Service Locator pattern for easy access across the app. Firebase Authentication safeguards user authentication processes, and Cloud Functions are used to manage serverless backend logic, like the queue prioritization algorithm. This setup aligns with the Microservices architecture, where each service is responsible for a specific functionality, ensuring scalability and ease of maintenance.

Integration with third-party APIs for additional functionalities, like medical information retrieval or payment processing, adheres to the Adapter pattern, allowing the app to interface seamlessly with external systems without affecting the core architecture. Security and compliance are paramount, with HTTPS for secure data transmission and adherence to healthcare regulations like HIPAA.

3.3.2. Database Design

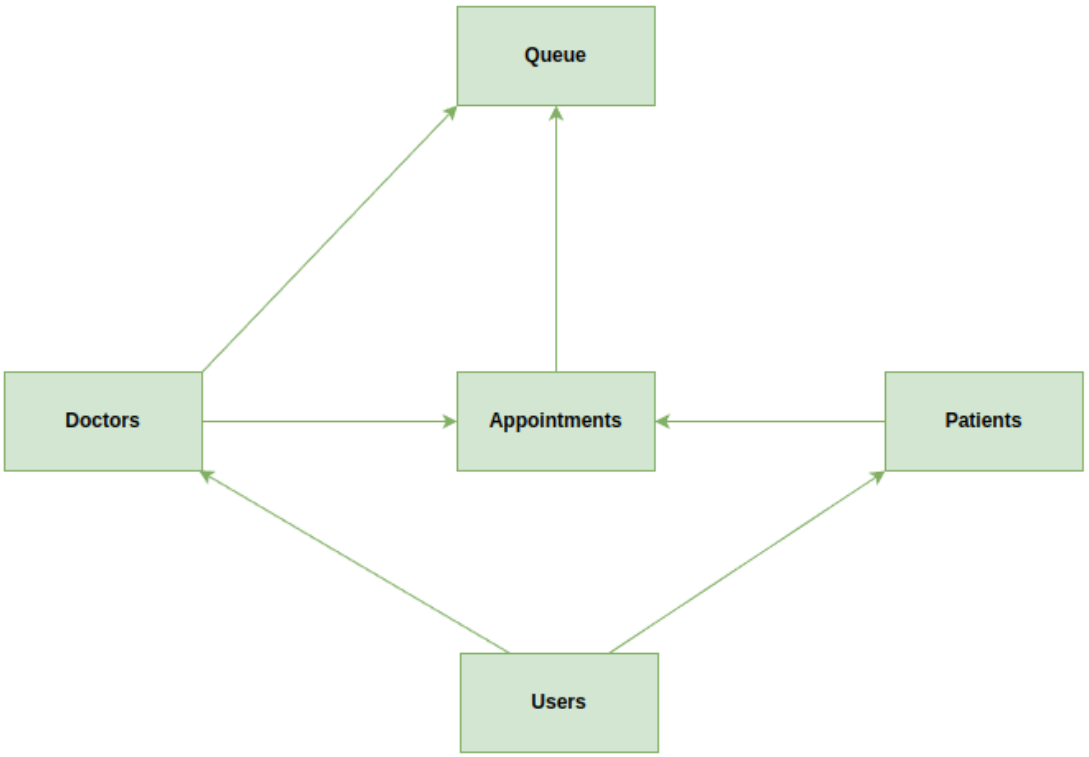

Figure 3.5. Database Design

The graph (Figure 3.5) visually represents the Firebase database structure for the Mobiheal application, delineating the interconnections among various data entities. At the core of this structure is the "Users" node, signifying a central collection that holds essential information about all users of the app, which includes both patients and doctors. This centralization facilitates efficient management and retrieval of user-related data.

Branching out from the "Users" node are two pivotal nodes: "Doctors" and "Patients." These nodes represent specialized collections that store additional, role-specific information. The "Doctors" node contains details pertinent to medical professionals, such as their

specializations and availability, while the "Patients" node holds individual health data and prescription records. This segregation ensures that data relevant to each user type is organized and easily accessible.

The "Appointments" node is another critical component of the database, linked to both "Doctors" and "Patients." It represents a collection that manages the scheduling and status of appointments. This arrangement demonstrates how appointments are a point of convergence for both doctors and patients, encapsulating the interactions between these two user groups within the application.

Additionally, the "Queue" node, which is a subordinate node of the "Doctors" and directly related to the "Appointments," is essential to the organization and prioritization of patient consultations. It displays the creative way in which the application manages patient lines, giving them priority according to many factors instead of just first-come, first-served.

Here is the structure of each entity inside the Database Model (NoSQL Database):

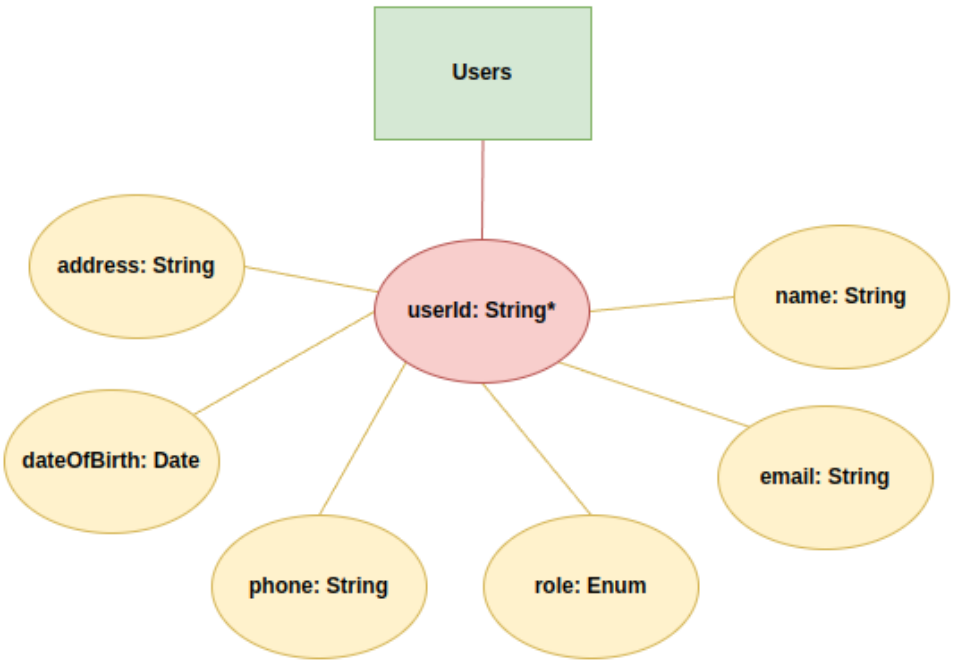

Figure 3.5.1. 'Users' Collection

The 'Users' collection (Figure 3.5.1) is the foundational model of the database, serving as a central repository for all user data. Each record, identified by a unique 'userId', stores essential information such as the user's name, email, role (distinguishing between patients and doctors), phone number, date of birth, and user address. This model is pivotal for authentication and user profile management within the Mobiheal.

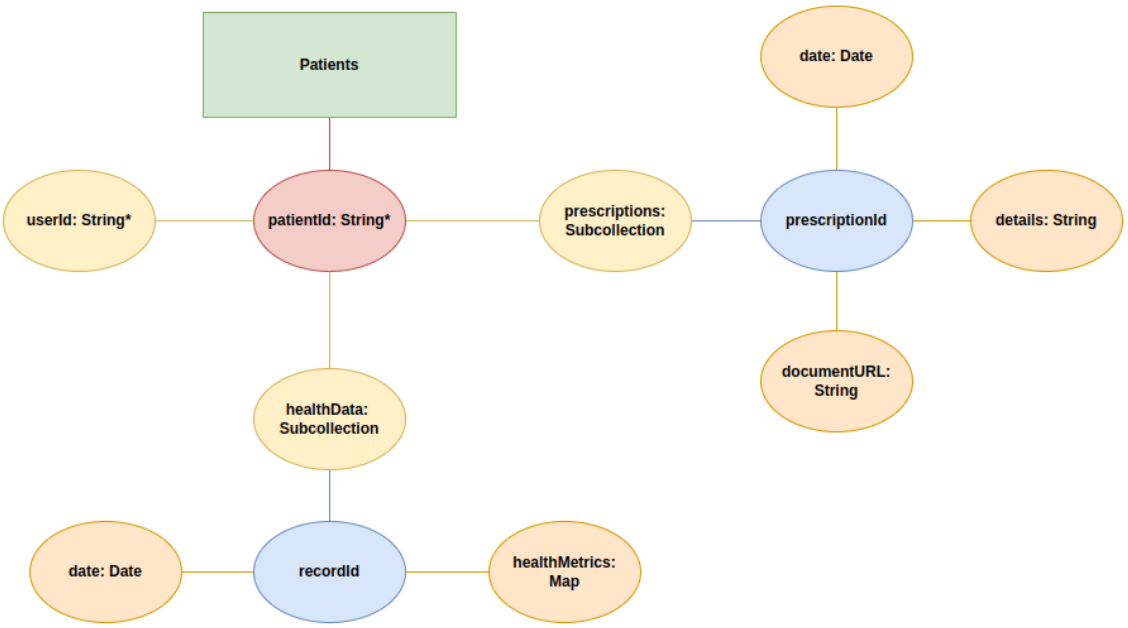

Figure 3.5.2. 'Patients' Collection

In the 'Patients' collection (Figure 3.5.2), each record is linked to a 'userId' and is dedicated to storing patient-specific data. It includes subcollections for health data and prescriptions, allowing patients to store and manage their health metrics and medical prescriptions over time. The 'healthData' subcollection tracks various health metrics with timestamps, while the 'prescriptions' subcollection keeps a record of digital copies of prescriptions, complete with details and a link to the document. This model is essential for maintaining comprehensive patient health records within Mobiheal.

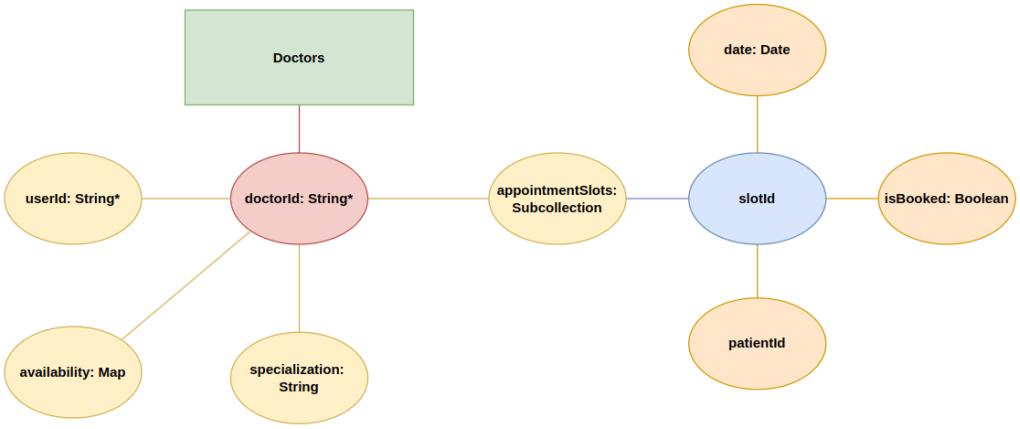

Figure 3.5.3. 'Doctors' Collection

The 'Doctors' collection (Figure 3.5.3) is tailored specifically for medical professionals using the app. Linked to the 'Users' collection through 'userId', it contains details pertinent to a doctor's practice, such as their specialization and availability. A critical feature of this collection is the 'appointmentSlots' subcollection, which holds the individual appointment slots with their date, booking status, and the patient's reference if the slot is booked. This model facilitates the management of doctor schedules and appointments.

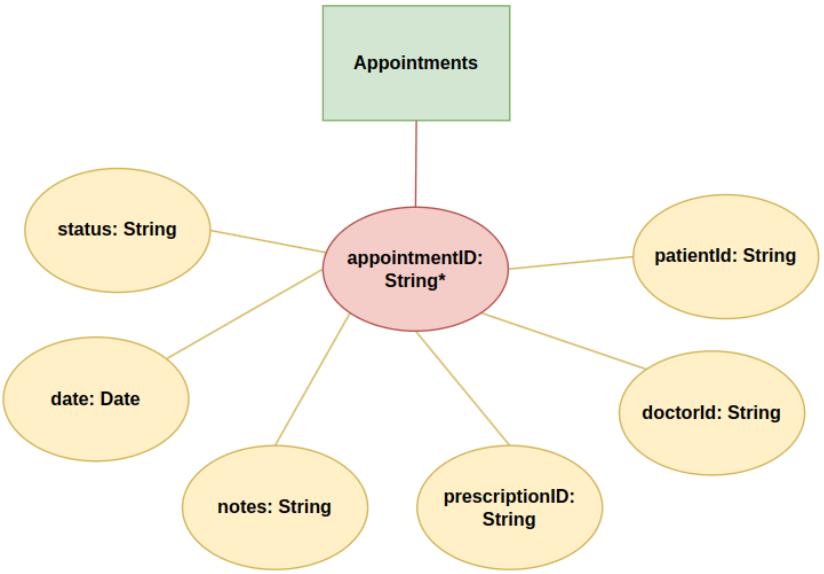

Figure 3.5.4. 'Appointments' Collection

The 'Appointments' collection (Figure 3.5.4) is a crucial model for managing the interactions between patients and doctors. Each appointment record, identified by 'appointmentId', holds references to both the patient and the doctor involved, along with the appointment date and status (scheduled, completed, or canceled). It can optionally link to a prescription from the patient's records and also allows doctors to add notes. This model is central to the scheduling and tracking of appointments in the app.

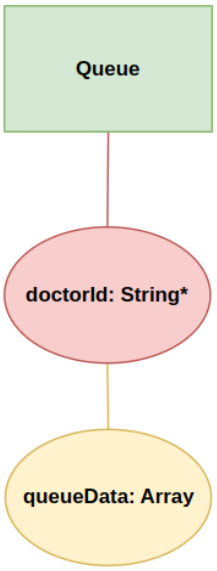

Figure 3.5.5. 'Queue' Collection

The 'Queue Management' collection (Figure 3.5.5) is designed to handle the prioritization and organization of patient appointments for each doctor. Linked to each doctor, it uses an array within 'queueData' to reference appointments from the 'Appointments'

collection. These references are sorted based on the app's prioritization algorithm. This model is key to ensuring that patients are seen in an order that optimizes care and efficiency.

3.3.3. User Interface Design

The usability and user experience may be significantly improved by a well-designed user interface. Thus, by combining contrasting colors and functional layouts with adaptable design, visual signals, and feedback messages, this system offers a straightforward but user-friendly online application.

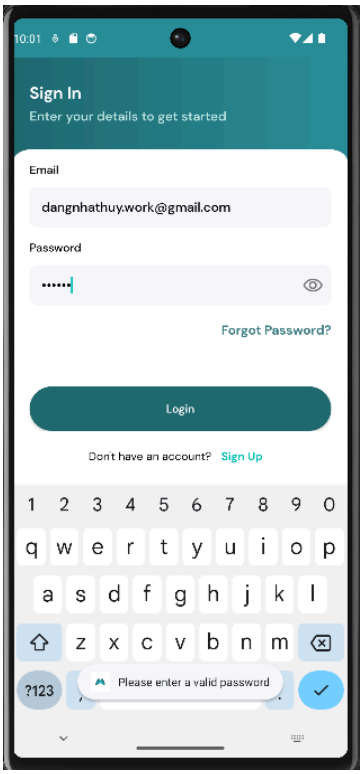

Figure 3.6. Concise And Informative Feedback Messages

The feedback messages (Figure 3.6) are strategically designed to guide users through their interactions with the app, providing clear and immediate feedback on actions such as form submissions, data entries, and error notifications. This approach not only helps reduce user errors but also significantly improves the overall user experience by making the app more interactive and responsive to user actions.

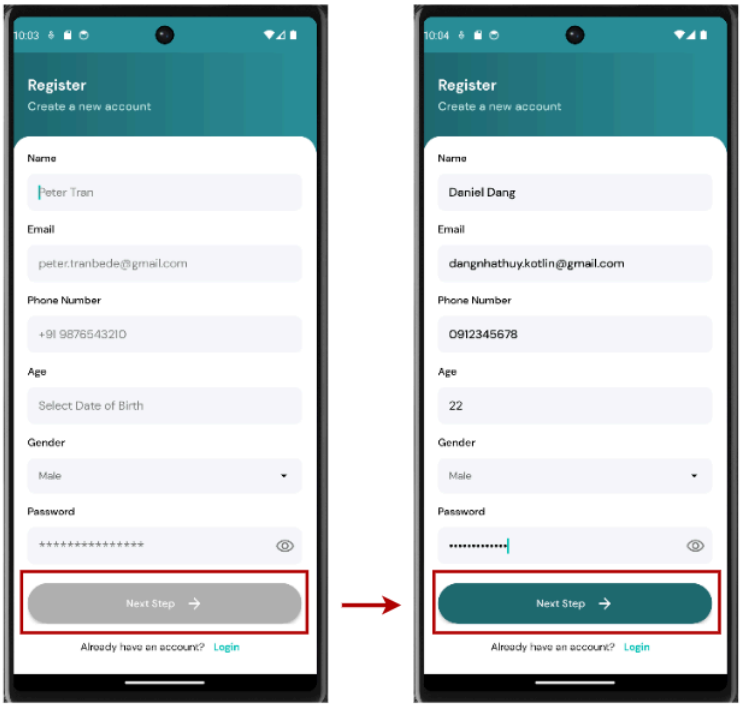

Figure 3.7. The Button Changes Color When Filling

The dynamic color change of the button element (Figure 3.7) is particularly effective in form interfaces, where buttons change color once all the required information is filled in. This visual cue serves as an intuitive indicator for users, confirming that they have completed the necessary fields and can proceed with their intended action, such as submitting a form or finalizing an appointment booking.

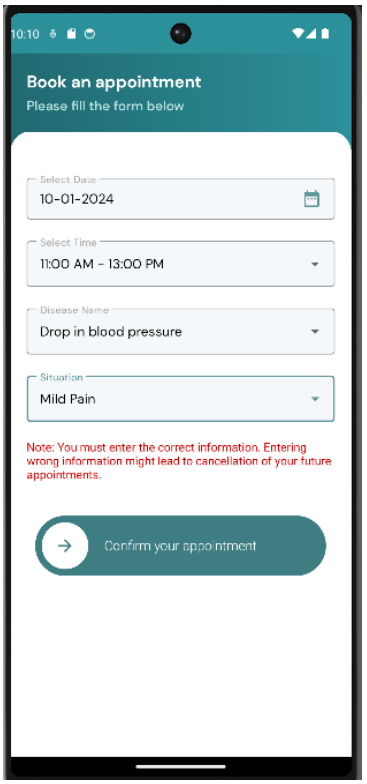

Figure 3.8. Drag The Slider To Confirm

Additionally, Mobiheal incorporates an innovative drag-to-confirm slider (Figure 3.8), which introduces an interactive element to the confirmation process. Instead of the traditional tap-to-confirm approach, users can drag a slider across the screen to finalize their actions. This not only adds a layer of engagement to the app but also serves as a functional design choice, reducing accidental confirmations and ensuring that users consciously complete important actions within the app.

3.3.4. Flow Of Application

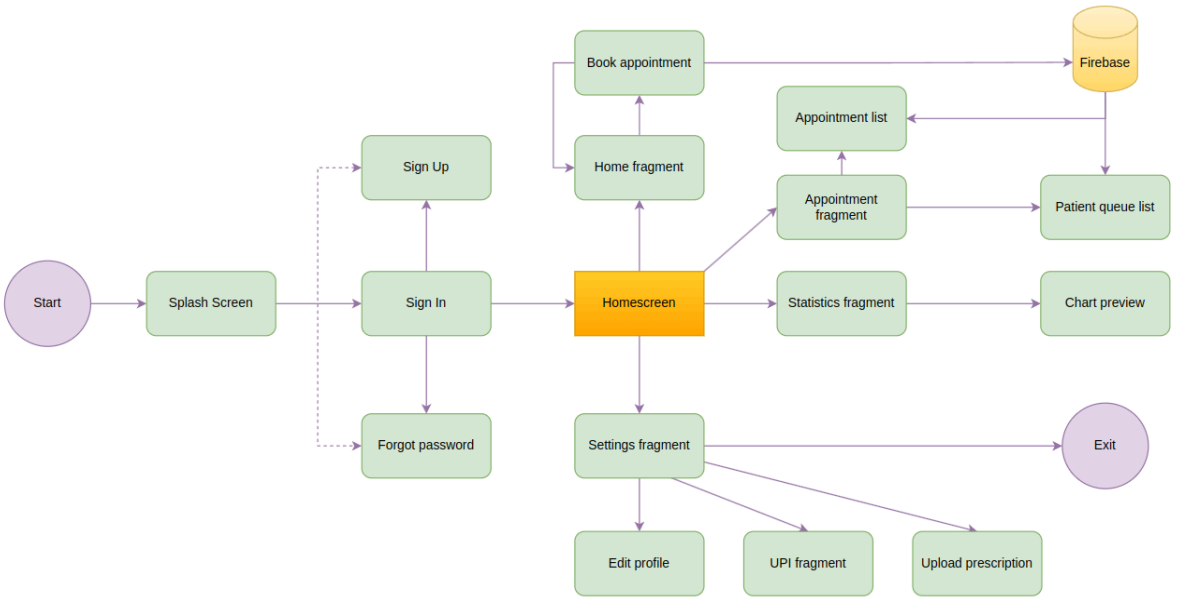

Figure 3.9. Mobiheal's Flow

As shown in Figure 3.9, begins with the user encountering a splash screen upon starting the app. This initial screen transitions to authentication options where the user has the choice to either sign in to an existing account or sign up to create a new one. There's also a provision for users who have forgotten their passwords to recover their accounts.

Once authenticated, the user lands on the home screen, which serves as the central hub for navigation. From here, the user has multiple pathways:

1. Booking Appointments: The user can book an appointment, which involves interacting with the database—presumably Firebase—to select a doctor and schedule a time. This process updates the appointment list that tracks all scheduled appointments.
2. Viewing Appointments: The user can navigate to the appointment fragment to view their list of appointments. This section also connects to the Firebase database, allowing the user to view and manage their appointments. Additionally, the user can access the patient queue list, giving them visibility into their position in the queue and expected wait times.
3. Health Statistics: Another pathway from the home screen leads to the statistics fragment, where the user can view their health statistics. This is likely a graphical

representation, such as a chart preview, that visually displays the user's health data over time for easy comprehension and tracking of health trends.

4. Settings and Personalization: The user can access a settings fragment from the home screen to customize their app experience. This section includes options to edit their profile or manage other app settings.
5. Payments and Prescriptions: Additionally, users have the option to navigate to the UPI fragment to handle payments, which would interact with the payment gateway to process transactions. They can also upload their prescription, enhancing their profile with the latest medical information for doctors to review.

3.4. Development Process: Agile Methodology

We are designed to carry out a thorough and accurate development process utilizing the Agile Methodology, particularly the Scrum framework. (Salah)

It is easier to keep track of the client's requirements and the project scope when employing a lightweight development approach like Agile. Even though there is no set timeline for the development process, customer and user demands can be assessed or refactored after every cycle to increase efficiency and save time.

Our team consists of the Production Owner, the Scrum Master (Team Leader), and the Developers. We will all act as the Scrum Master at every level, which is a special feature of our methodology. Every team member can get fresh insights and increase their commitment to the project.

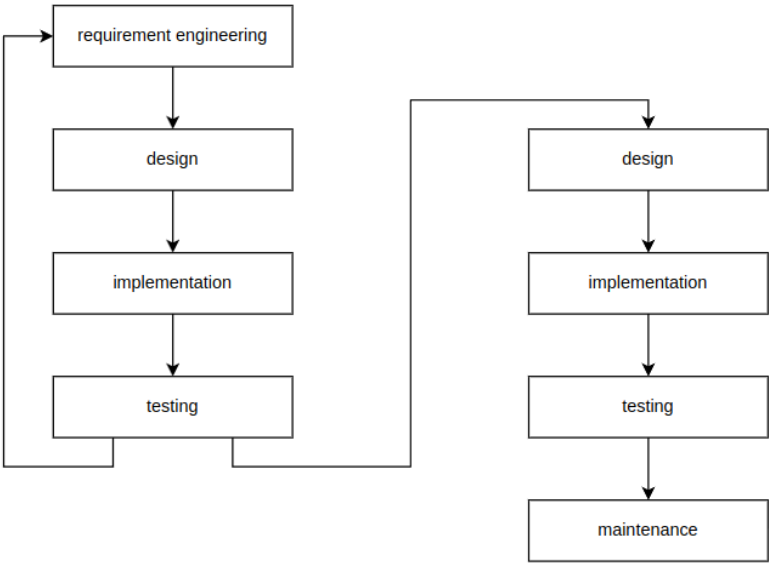

Figure 3.10. Evolutionary Prototyping

To create the initial version of the system, the client and user first define the fundamental needs. This prototype undergoes several revisions until the final version (also known as evolutionary prototyping) is completed.

CHAPTER 4

IMPLEMENT AND RESULTS

4.1. Logic implementation

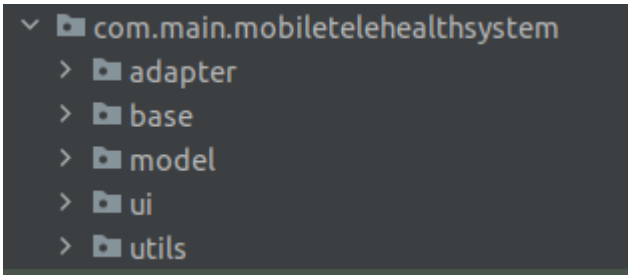

Figure 4.1. Overall Structure Of Application

The core package *com.main.mobiletelehealthsystem* hints at its purpose. Subdirectories like *adapter*, *model*, and *ui* separate key functionalities: data display, underlying data structures, and user interface components. "base" likely hold foundational elements, while "utils" within "ui" suggests reusable interface elements.

4.1.1. Adapter

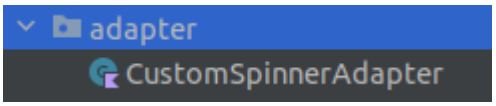

Figure 4.2. 'adapter' Package

CustomSpinnerAdapter is a versatile Android *RecyclerView* adapter designed to display lists of strings with unique click behavior for each item. It stores data in a mutable list, leverages a custom *ViewHolder* to manage item views efficiently, and offers key functions for item creation, binding, data set management, and notification handling. The *setItem* function enables seamless updates to the adapter's data and ensures smooth UI updates by notifying the adapter of changes.

4.1.2. Base

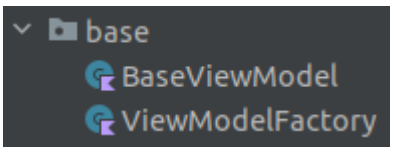

Figure 4.3. 'base' Package

ViewModelFactory is a crucial class for managing the lifecycle and creation of *ViewModel* instances, ensuring data consistency and adherence to best practices. It enables custom *ViewModel* constructors, such as those requiring an Application object, and plays a central role in instantiating various *ViewModel* subclasses (*HomeViewModel*, *SignUpFirstViewModel*, etc.) that are responsible for handling data and logic within specific application features.

4.1.3. Model

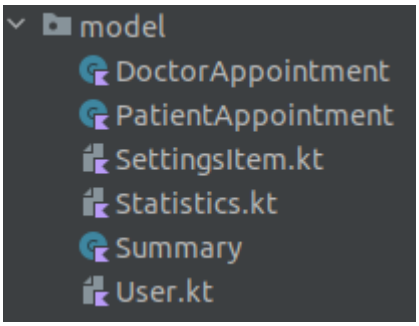

Figure 4.4. 'model' Package

By implementing the *MVVM* architecture, we ensure a well-structured and maintainable codebase. *BaseViewModel.kt* provides a foundation for *ViewModel* classes, offering a progress indicator for UI feedback. *ViewModelFactory.kt* manages *ViewModel* creation, guaranteeing proper lifecycle handling. *SignInRepository.kt* encapsulates sign-in data operations, including Firebase Authentication. Data model classes (*PatientAppointment.kt*, *Summary.kt*, etc.) define clear structures for diverse data types, promoting organization before storage or UI display.

4.1.4. Utils

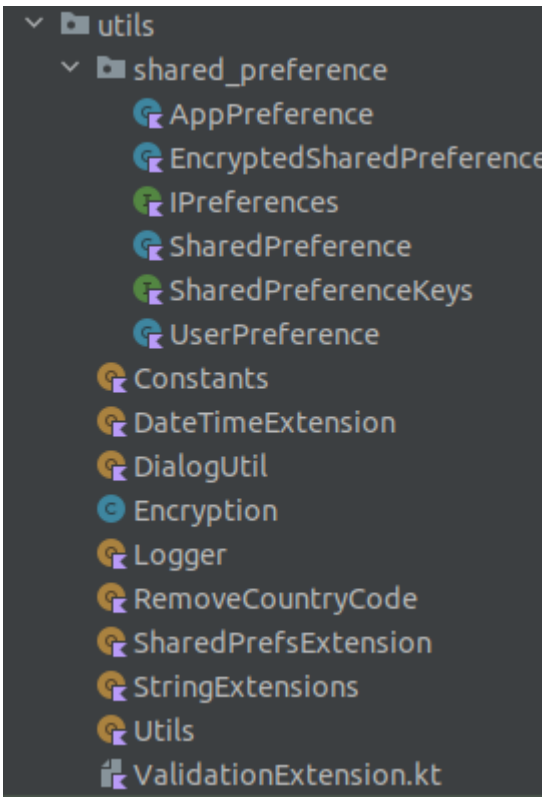

Figure 4.5. 'utils' Package

Data model classes like *Statistics.kt* define health-related data structures, while *SignInRepository.kt* handles sign-in operations using Firebase Authentication. *User.kt* represents user data, and utility classes like *Constants.kt*, and *DateTimeExtension.kt*, *DialogUtil.kt*, and *logger.kt* offers helpful functions for constants, date/time handling, dialogs,

and logging. *Encryption.java* provides encryption/decryption capabilities using AES, ensuring data security.

4.1.5. UI

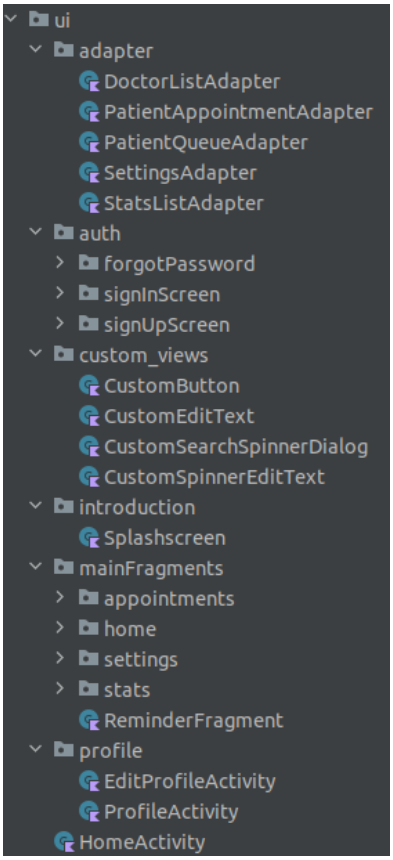

Figure 4.6. 'ui' Package

a. Adapter:

These files define crucial data structures and UI components, ensuring effective user management, settings handling, and information display. *User.kt* models system users (patients or doctors) and their assessments. *SettingsItem.kt* represents settings items. *SignInViewModel.kt* handles sign-in logic. *PatientAppointmentAdapter.kt*, *PatientQueueAdapter.kt*, *SettingsAdapter.kt*, *StatsListAdapter.kt*, and *DoctorListAdapter.kt* are *RecyclerView* adapters, each tailored to display specific lists in the UI, such as appointments, patient queues, settings, health statistics, and doctors, respectively.

b. Auth

These files collectively define data structures and manage essential user-related features, ensuring a well-organized and functional codebase. *Statistics.kt* models health data, *User.kt* represents users and *SettingsItem.kt* defines settings items. *ForgotPasswordActivity.kt* and *ForgotPasswordViewModel.kt* handle password resets, while *SignInScreen.kt*, *SignInViewModel.kt*, and *SignInRepository.kt* manage sign-in functionality using Firebase authentication. *SignUpFirstScreen.kt*, *SignUpFirstViewModel.kt*, *SignUpSecondScreen.kt*, *SignUpSecondViewModel.kt*, and *SignUpRepository.kt* work together to provide sign-up processes, also utilizing Firebase. *DoctorListAdapter.kt* is a RecyclerView adapter specifically designed to display lists of doctors within the UI.

c. Custom Views

These files collectively define data structures, UI components, and *ViewModel* logic, ensuring a well-organized and user-friendly experience. *User.kt* models users and their attributes, while *SettingsItem.kt* represents settings items. *SignInViewModel.kt* handles sign-in logic, including authentication, data retrieval, and storage. CustomButton.kt, *CustomEditText.kt*, *CustomSearchSpinnerDialog.kt*, and *CustomSpinnerEditText.kt* provide tailored UI elements for enhanced user interactions. *DoctorListAdapter.kt* is a *RecyclerView* adapter specifically designed to display lists of doctors.

d. Introduction

Splashscreen.kt serves as the application's initial welcome screen, providing a visual placeholder while crucial setup tasks are completed. It first initializes Firebase Authentication and checks for a current user. If a user is already logged in, it smoothly directs them to the *HomeActivity* after a 2-second delay for a seamless experience. If no user is logged in, it guides them to the *SignInScreen* activity, ensuring appropriate authentication before accessing the main app features.

e. Main fragment

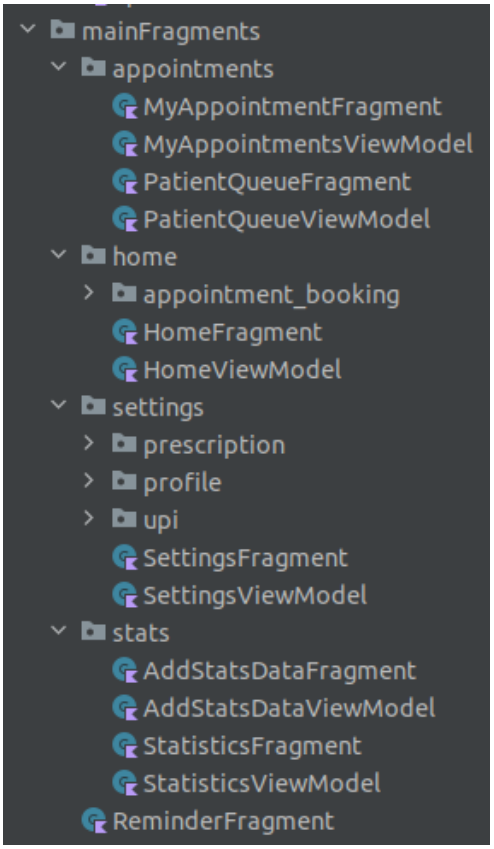

Figure 4.7. 'mainFragments' folder

The *Appointment* section (including *ReminderFragment.kt*) handles reminder management and display, enabling users to create, view, and manage their reminders effectively. The *Home* section houses the primary home screen components, providing the main UI and features users interact with upon opening the app, potentially including appointment scheduling functionality. The *Settings* section manages app settings and user preferences, covering aspects like user profiles, prescription settings, and potentially UPI payment options. The *Stats* section focuses on organizing and presenting app statistics,

incorporating statistical data and analytics to enhance the app's data-driven elements and provide users with analytical insights.

f. Profile

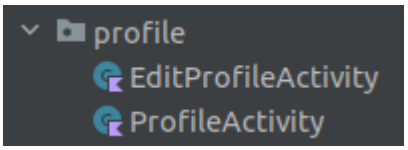

Figure 4.8. 'profile' folder

Splashscreen.kt handles the app's first impression, while *SettingsFragment.kt* empowers users to tailor their experience. *Splashscreen.kt* acts as a visual placeholder while setup tasks are completed, checking for a current user and seamlessly directing them to either *HomeActivity* or *SignInScreen* based on their login status. *SettingsFragment.kt* offers a variety of options for users to manage their app preferences, including UPI settings, prescription submissions, profile editing, feedback submission, help access, and logout functionality, ensuring a personalized and adaptable experience.

g. MainActicity

```
class HomeActivity : AppCompatActivity() {

    private lateinit var _binding: ActivityHomeBinding
    private var timer = 0L

    Nguyên Bê Đê
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        _binding = ActivityHomeBinding.inflate(layoutInflater)
        setContentView(_binding.root)

        //Hides action bar
        supportActionBar?.hide()

        val bottomNavigationView = _binding.bottomNav
        val navController: NavController = findNavController(R.id.fragmentContainerView)
        AppBarConfiguration(setOf(R.id.home, R.id.stats, R.id.appointment, R.id.settings))

        bottomNavigationView.setupWithNavController(navController)

        navController.addOnDestinationChangedListener { _, destination, _ ->
            when (destination.id) {
                R.id.home -> showBottomNav(bottomNavigationView)
                R.id.stats -> showBottomNav(bottomNavigationView)
                R.id.appointment -> showBottomNav(bottomNavigationView)
                R.id.settings -> showBottomNav(bottomNavigationView)
                else -> hideBottomNav(bottomNavigationView)
            }
        }
    }
```

Figure 4.9. HomeActivity Class

The application's primary activity, *HomeActivity.kt*, is in charge of overseeing the fragments that go along with the bottom navigation view. The activity's layout is inflated, the *NavController* is configured with the *BottomNavigationView*, and a listener is added to the *NavController* so that the *BottomNavigationView* may be shown or hidden depending on the current destination. To handle the back press action, it additionally overrides the `onBackPressed` function.

4.2. Results

4.2.1. User Authentication

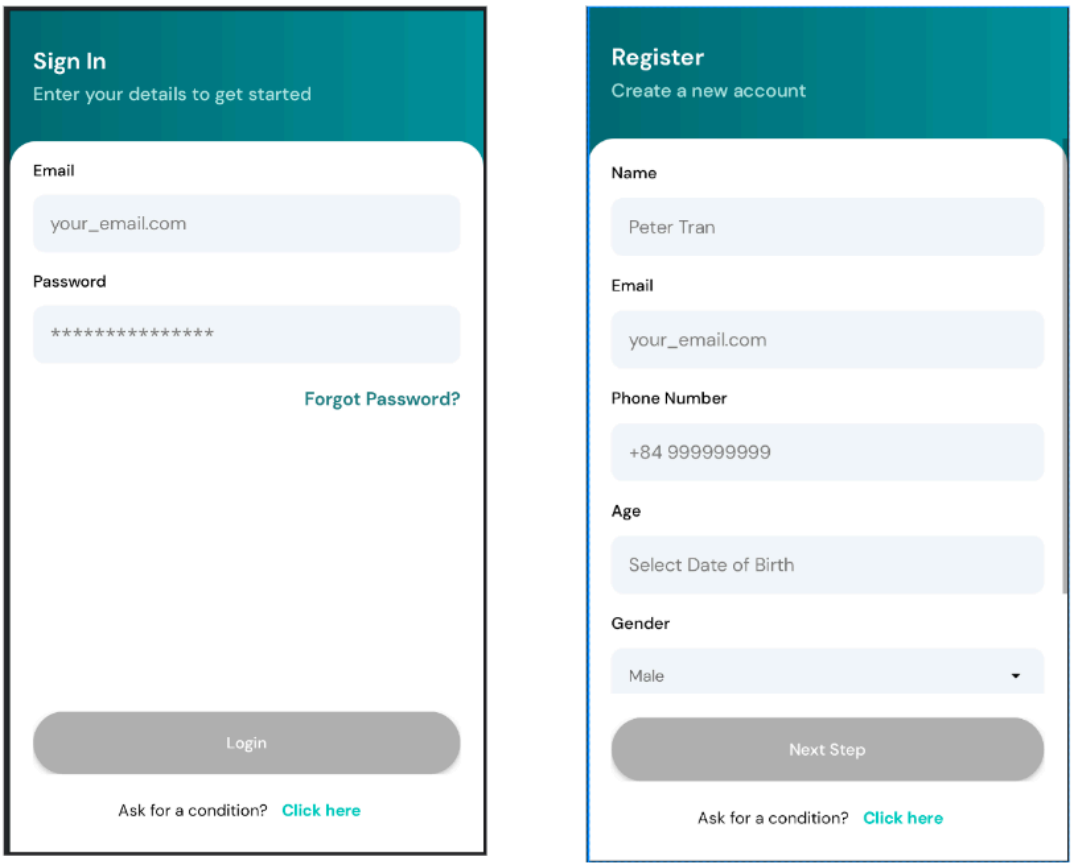

Figure 4.10. Register UI

We have two sections of the User Authentication, one for entering your email address and password to sign in, and the other for creating a new account (Figure 4.10).

The sign-in section is at the top of the screen and has fields for entering the user's email address and password. There is also a "Forgot Password?" link if the user has forgotten their password. Below the sign-in section is the register section. This section has fields for entering their name, email address, phone number, age, date of birth, and gender. There is also a button to select gender as male or female. Once they have entered all of their information, they can click the "Next Step" button to create their account.

4.2.2. User Profile

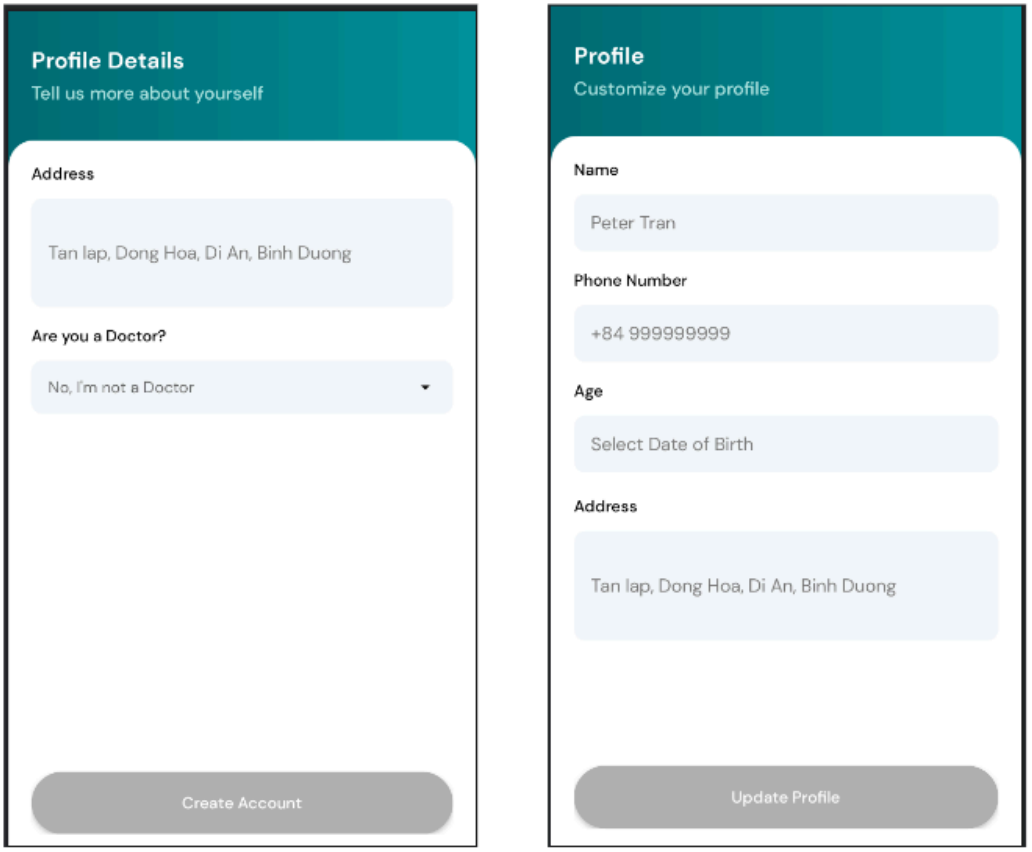

Figure 4.11. Profile Details & Edit Profile UI

The top section features the app's logo and the text "Profile Details". Below, a prompt reads "Tell us more about yourself" alongside a button to "Customize your profile", inviting users to personalize their experience (Figure 4.11).

Central to the page are sections for key user details, including address, name, phone number, and a doctor verification prompt. The address field is pre-filled, showcasing potential integration with existing user data. Other sections are currently blank, with placeholder text and edit buttons encouraging users to input their information. Buttons for creating an account and updating the profile are prominently placed, granting users control over their data.

4.2.3. Appointment Booking

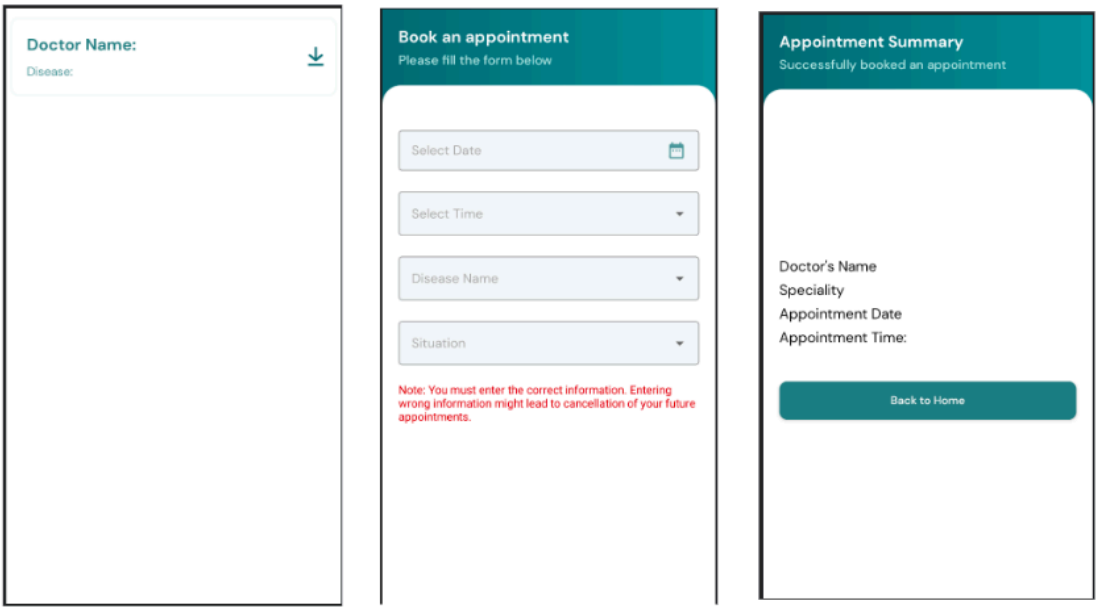

Figure 4.12. Appointment Booking Pages

The top section displays the doctor's name and a button to book an appointment (Figure 4.12). Below, two sections titled "Disease" and "Situation" are outlined. The "Disease" section prompts users to fill out a form, while the "Situation" section offers a free text field for further details.

A calendar icon beside the "Disease" section suggests the possibility of scheduling appointments based on specific dates. However, the current date and available time slots are not visible in the screenshot. Additionally, a note warns users about the importance of accurate information, implying potential consequences for incorrect entries.

4.2.4. Statistic

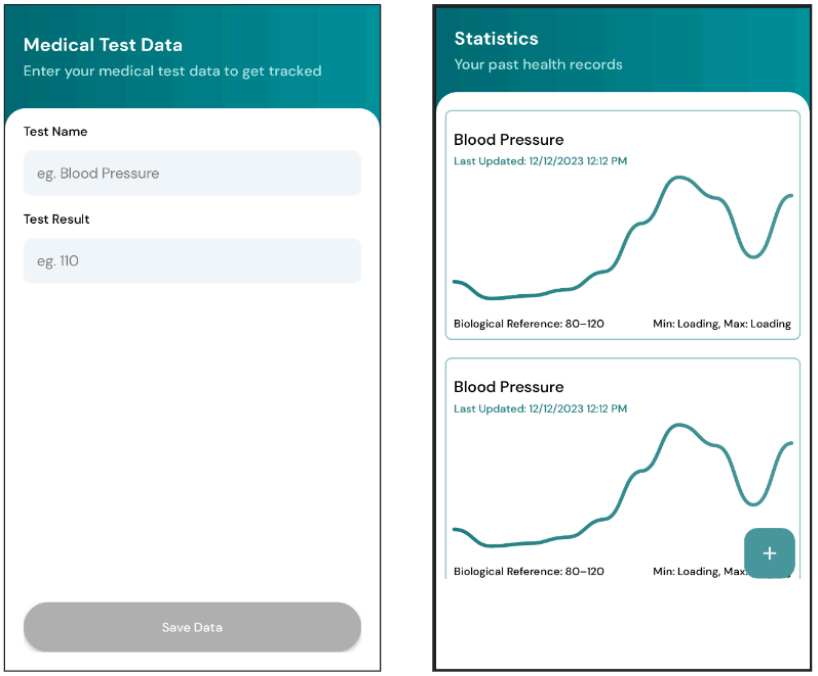

Figure 4.13. Statistics Pages

The top section displays the app's logo and the text "Medical Test Data" with a "Statistics" tab to the right (Figure 4.13). Below, a prompt invites users to "Enter your medical test data to get tracked," encouraging them to input their health information for monitoring.

The central section showcases a blood pressure test result, presumably entered by the user. It displays the systolic and diastolic readings along with the date and time. Notably, the reference range for healthy blood pressure is also shown, allowing users to compare their readings to standard values. A green plus sign button positioned next to the blood pressure reading suggests the option to add more data points, potentially enabling users to track their blood pressure over time.

4.2.5. Reminder

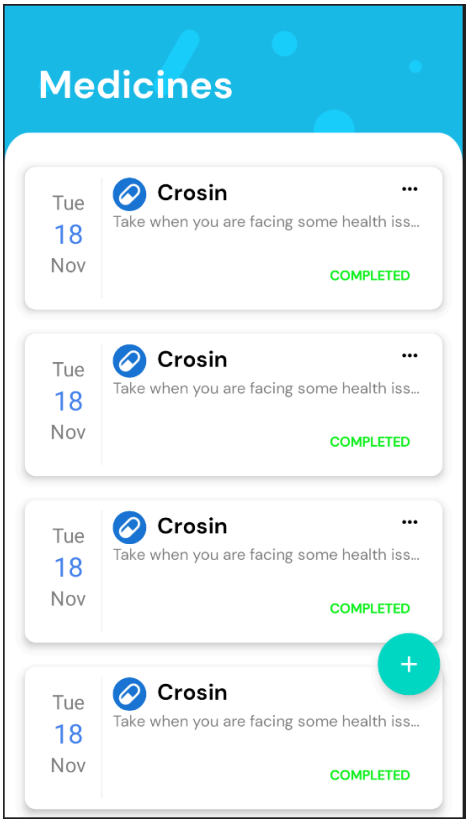

Figure 4.14. Medicines Reminder Page

The top section displays the app's logo and the date, for example, "Tuesday, January 2nd". Below, a prominent blue bar highlights the current time, for instance, "10:02 AM."

Beneath the time, a medication schedule fills the screen, organized by day and time. Each entry specifies a medication name, dosage, and instructions, with tick boxes to mark when each medication has been taken. Completed entries are indicated by a greyed-out appearance and a checkmark in the corresponding tick box.

4.2.6. Rating

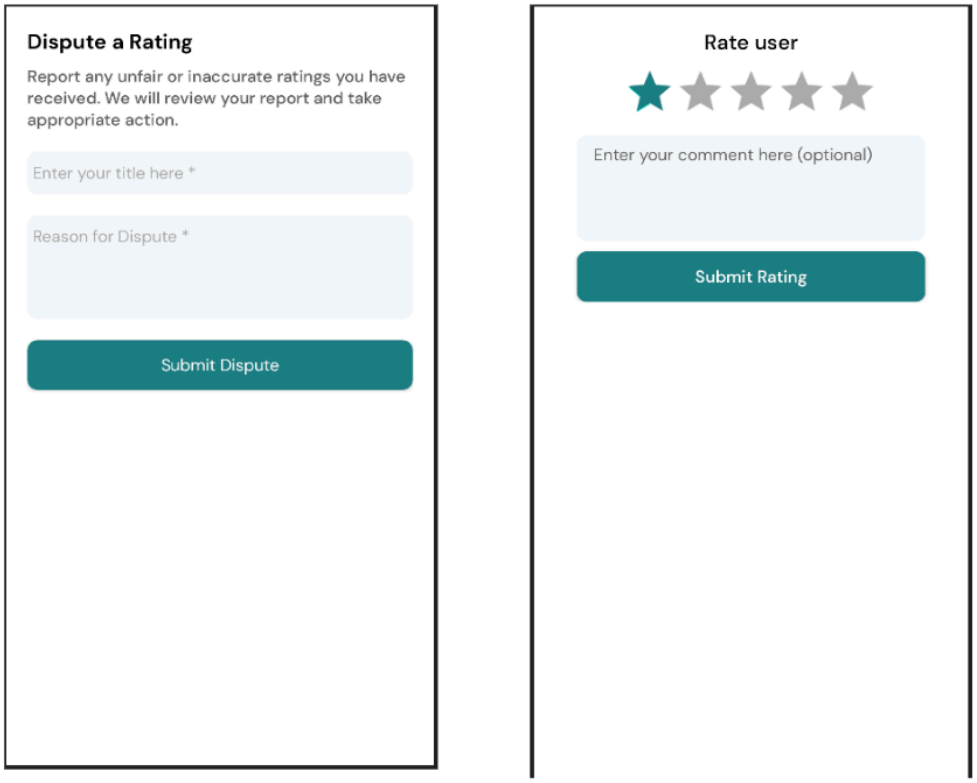

Figure 4.15. Rating Pages

The top section displays the title "Dispute a Rating" in bold text, followed by a prompt to "Report any unfair or inaccurate ratings you have received." Below this, there are two distinct sections: one for reporting the rating and another for rating the user who left the disputed rating.

The "Report a Rating" section has fields for entering a title and an optional comment, along with a dropdown menu to choose the reason for the dispute. The reason options include "Factually inaccurate," "Biased or unfair," "Spam or offensive," and "Other." There's also a text box for providing additional details about the dispute. The "Rate User" section exists with a five-star rating system and a submit button.

CHAPTER 5

DISCUSSION AND EVALUATION

5.1. Discussion

In a world striving for universal healthcare, MobiHeal emerges as a beacon of hope, shattering the traditional mold of healthcare delivery. This mobile telemedicine app, crafted on the robust foundations of Firebase and Kotlin, is poised to radically transform the landscape, tackling age-old issues that have plagued patients for far too long.

Imagine a future where lengthy wait times and administrative burdens melt away, replaced by streamlined appointment scheduling, intelligent queue management, and the ease of digital prescriptions. Picture patients empowered, not passive participants, actively engaged in their healthcare journey through real-time communication, data visualization tools, and personalized care options tailored to their unique needs. This is the transformative power of MobiHeal, making once-distant specialists readily accessible through telemedicine consultations, regardless of one's geographical location or physical limitations.

MobiHeal understands that healthcare is not a one-size-fits-all endeavor. Its revolutionary spirit extends beyond technological brilliance, embracing a deep commitment to inclusion and fairness. It levels the playing field, ensuring everyone, regardless of their background or physical constraints, has the right to quality healthcare. Culturally sensitive care bridges the gap between medical expertise and local traditions, while multilingual support fosters effective communication, breaking down linguistic barriers. Through partnerships with local groups and community leaders, MobiHeal empowers these crucial stakeholders to become change agents, raising awareness about health concerns, promoting preventative care, and building lasting healthcare solutions within their communities.

This is not merely about bridging existing gaps; it's about boldly reimagining the very fabric of healthcare delivery. MobiHeal envisions a future where healthcare is no longer confined to the sterile walls of hospitals or dictated by the tyranny of distance. It envisions a future where healthcare is mobile, adaptable, and personalized, flowing freely to meet the needs of individuals and communities. It's a future where a doctor's reassuring touch can reach through a smartphone screen, and the hum of a wearable device whispers health insights in real-time.

MobiHeal is not just an app; it is a revolution in the making. It's a clarion call for a healthcare system that prioritizes equality, accessibility, and inclusivity. It's a promise whispered on the wind, carried by the wings of technology, that one day, healthcare will be a fundamental right, not a privilege reserved for the fortunate few. It's a future where everyone, everywhere, can experience the transformative power of good health, and MobiHeal stands as the vanguard, leading the charge toward a healthier, more equal world for all.

5.2. Evaluation

5.3.1. Self-evaluation

- The team is committed to submitting the final report punctually.
- We will guarantee the completion and smooth functioning of the essential elements and operations of the database, ensuring it operates without errors.
- To monitor progress, team members will use Trello to regularly check and verify the work completed by the assigned member after each phase.
- As the integration of the database application is not mandatory, its strategy and process will be outlined in the Final Report, should its development occur.
- The Final Report will include an in-depth explanation of the technologies used in the project's research.

5.3.2. Teacher’s evaluation

- Deadline: 11:59 p.m. on January 15th, 2022.
- The project will be submitted to the Professor.
- Points will be awarded based on the state of the Final Report and the project's grading standards, which the Professor determines.

CHAPTER 6

FUTURE CHALLENGES AND CONCLUSION

6.1. Future Challenges

Presently, the existing development is sufficient for basic real-life tasks. However, the market is rapidly evolving, especially in advanced UI design, aimed at attracting a larger user base and fostering business collaborations. This evolution contributes to the prosperity of both the design and development teams. It's not just about the UI design; the backend functionality of each feature will also be crucial in addressing and overseeing issues. This aspect is responsible for retrieving data when both the user and doctor are utilizing the application. Furthermore, security is a primary concern when developing a secure application. (Agency)

Data Encryption: Ensure that sensitive data, such as user credentials, is encrypted both during transmission and storage. Implementing secure encryption protocols, like TLS for communication and strong encryption algorithms for storage, helps prevent unauthorized access.

Authentication and Authorization: Review and reinforce the application's authentication mechanisms. Enhance user authentication processes with secure practices like multi-factor authentication. Additionally, carefully manage user permissions and implement robust authorization mechanisms to control access to different parts of the application.

Secure Data Storage: If the enhancements involve changes to data storage, consider secure storage practices. Avoid storing sensitive information in plain text, and utilize secure storage options provided by the Android platform, such as the Keystore system.

Input Validation: Strengthen input validation mechanisms to prevent common security vulnerabilities like SQL injection, cross-site scripting (XSS), and other injection attacks. Ensure that user inputs are validated and sanitized before being processed.

6.2. Conclusion.

In conclusion, while the current development adequately serves basic real-life tasks, the dynamic nature of the market necessitates a swift adaptation to advancements, particularly in advanced UI design. This evolution not only benefits the design and development teams but also underscores the importance of robust backend functionality in addressing various issues. The seamless retrieval of data, especially in scenarios involving users and doctors, is a critical aspect of application performance.

Moreover, the enhancement of an application goes hand in hand with addressing security concerns. Key measures include data encryption to protect sensitive information during transmission and storage, reinforcing authentication mechanisms with multi-factor authentication, adopting secure data storage practices, and implementing rigorous input validation to prevent common security vulnerabilities.

By incorporating these security measures alongside continuous advancements in UI design and backend functionality, developers can ensure a comprehensive and secure user experience. As the landscape of technology evolves, a proactive approach to both functionality and security will be integral to the success and longevity of any application in the dynamic marketplace. (10 Common Challenges)

REFERENCES

"10 Common Challenges of Android App Development." *Fullestop Blogs*, 5 Jan. 2024, www.fullestop.com/blog/what-are-10-common-challenges-most-android-developers-encounter#Conclusion.

Agency, DECODE. "8 Biggest Challenges of Android App Development." *DECODE*, 4 Sept. 2023, decode.agency/article/android-app-development-challenges/.

Guest. "The Feasibility of Mobile Telehealth Solutions: Healthcare It Today." *Healthcare IT Today | Fresh, Daily, Practical Healthcare IT Insights*, 18 Jan. 2021, www.healthcareittoday.com/2020/10/07/the-feasibility-of-mobile-telehealth-solutions/.

Hospital, Tâm Anh. "Tâm Anh Hospital." *Bệnh Viện Đa Khoa Tâm Anh | Tâm Anh Hostpial*, 6 Jan. 2024, tamanhhospital.vn/.

Salah, D, Paige, RF & Cairns, PA 2014, A systematic literature review for agile development processes and user centred design integration. in *18th International Conference on Evaluation and Assessment in Software Engineering, EASE '14, London, England, United Kingdom, May 13-14, 2014*. pp. 1-10. https://doi.org/10.1145/2601248.2601276

TIGCAL, JOMAR. *Simplifying Android Development with Coroutines and Flows: Learn to Use Kotlin Coroutines and... the Flow API to Handle Data Streams Asynchronously*. PACKT PUBLISHING LIMITED, 2022.

Varaksina, Svitlana. "Mind Studios." *Blog - Mind Studios*, Mind Studios, 26 Oct. 2023, themindstudios.com/blog/telemedicine-development/.

Varma. "ContinuousCare." *Continuouscare*, www.continuouscare.io/. Accessed 6 Jan. 2024.

APPENDIX