

A HEALTH- API AND MOBILE APPLICATION





By  
GROUP BSE 23-9  
Health Systems  
DEPARTMENT OF NETWORKS  
SCHOOL OF COMPUTING AND INFORMATICS TECHNOLOGY

A Project Report Submitted to the School of Computing and Informatics Technology  
For the Study Leading to a Project in Partial Fulfillment of the  
Requirements for the Award of the Degree of Bachelor of  
Software Engineering of Makerere University.

Supervisor  
Dr. Rashidah Namisanvu Kasauli  
Department of Networks  
School of Computing and Informatics Technology, Makerere University  
[Supervisors Email Address], +256-41-540628, Fax +256-41-540620 June, 2023.

**Declaration**

We, group BSE 23-9, hereby declare that the work presented is original and has never been submitted for an award to any university or institution of higher learning. We can confirm that where we have done consultations from either published material or the works of others, it has been attributed in this report.

#	Names	Registration Number	Signature
1	EMOKOL HILLARY	19/U/8797/EVE	
2	RUTAHIGWA EMMANUEL NOEL	19/U/12154/PS	
3	KARAMUZI GIFT JOSHUA	19/U/11186/EVE	
4	NGABIRANO ISAAC	19/U/12546/PS	

## **Approval**

This project report titled "A HEALTH-API AND MOBILE APPLICATION" has been submitted for examination with my approval as the supervisor of group BSE 23-9.

Dr. Rashidah Kasauli Namisanvu

Department of Networks

School of Computing and Informatics Technology;

College of Computing and Information Sciences,

Makerere University

Signature: ..... Date: .....

Supervisor

## **Dedication**

We dedicate this report to our supervisor, Dr. Rashidah Kasauli Namisanvu who has persistently guided and contributed to the success of this project.

Additionally, we dedicate this report to all the health professionals who strive to provide the best health services to the people of our respective communities. With this system in place, you can rest assured that you will receive timely medical services allowing you to respond to your symptoms with recommendations in a faster and easier way.

Finally, we dedicate this report to our loved ones who have supported us for the duration of this Project.

## **Acknowledgements**

First and foremost, we express our gratitude to the Almighty God for His guidance and support Throughout our degree program, and specifically during the process of writing this report We thank our supervisor, Dr. Rashidah Kasauli Namisanvu, for his invaluable guidance and support Throughout the research, process and we also thank Dr. Mary Nsabagwa for her helpful advice in addition, support.

We are deeply grateful to all of the participants in this study for their willingness to share their Experiences and insights with us. Your contributions have been invaluable to the success of this Report.

Finally, we thank ourselves, the team members, Joshua Gift Karamuzi, Emokol Hillary, Isaac Ngabirano and Rutahigwa Emmanuel Noel for our hard work and collaboration to enable the completion of this project.

## **Abstract**

This report presents the Health API and Mobile Application System that has been developed aiming at integrating multiple hospital databases across the country through the Application-programming interface. The services provided by this Api will then be consumed by two mobile applications with one of the application for health professionals which will help them determine an outbreak of a particular disease with the use of the statical graph and data, entering patient information in the database for storage and finally be able to view and edit patient medical report. The other mobile application will be on for the user(patient) , this application will have a chatbot feature that will provide patients with recommendations based on their communicated symptoms(to be specific we focus on malaria) using natural language processing and also it will provide automated AI powered reminders using generative AI for reminding patients about their prescription.

The implementation of this Health API and Mobile Application System not only brings peace of mind to patients but also offers great convenience. It enables patients to have a mobile doctor to ask for recommendations, which enables them to ask at their own convenience. It also helps health workers know their patient's details since the API can access them therefore; they have the ability to know the patient's medical history.

With the Health API and Mobile Application System, both patients and health professionals can have confidence and ease in the provision of health services provided by the system.

## Table of Contents

### Contents

Declaration .....	2
Dedication .....	4
Acknowledgements .....	5
Abstract .....	6
Table of Contents .....	7
List of Figures .....	9
List of Tables .....	10
Abbreviations/Acronyms .....	11
Chapter 1: Introduction .....	12
Chapter 2: System Specifications .....	13
2.1 Version of requirements and Version Control .....	14
2.2 Input .....	15
2.3 Output.....	15
2.4 Functionality .....	16
2.5 Limitations and safety .....	17
2.6 Default settings .....	17
2.7 Special requirements .....	18
2.8 Errors and alarms .....	19
Chapter 3: Design output .....	21
3.1 Implementation (coding and compilation).....	21
3.4 Documentation .....	22
Chapter 4: Inspection and testing.....	23
4.1 Introduction.....	23
4.2 Test plan and performance .....	24
4.2.1 Test objectives.....	25
4.2.2 Scope and Relevance of tests .....	27
4.2.3 Levels of tests.....	29
4.2.4 Types of tests .....	30
4.2.5 Sequence of tests .....	31
4.2.6 Configuration and calculation tests .....	34
4.3 Precautions .....	34
4.3.1 Anomalous conditions.....	34
4.3.2 Precautionary steps taken.....	34
Chapter 5: Installation and system acceptance test .....	35
5.1 Input files .....	35

5.2 Supplementary files.....	36
5.3 Installation qualification .....	36
Chapter 6: Performance, servicing, maintenance, and phase out.....	38
6.1 Service and maintenance.....	38
6.2 Performance and Maintenance.....	39
Chapter 7: Conclusion and Recommendations .....	43



## List of Figures

Figure 1. User registration .....	45
Figure 2 user login .....	46
Figure 3. Patient home screen .....	47
Figure 4. Notifications .....	48
Figure 5. Chat screen .....	49
Figure 6. Health worker chatscreen .....	50
Figure 7. Data Explorer Home screen.....	51
Figure 8. medical records.....	52
Figure 9. Prescmod screen .....	53
Figure 10. general settings .....	54

## List of Tables

Table 1 Design Details .....	22
Table 2 Inspection and Performance.....	23
Table 3 Test Plan.....	24
Table 4. Test Objectives.....	25
Table 5. Scope and Relevance .....	28
Table 6. Test Case summary .....	31
Table 7 Input Files .....	35
Table 8. Supplementary files .....	36
Table 9. Installation summary.....	36
Table 10. Procedural check.....	37
Table 11. Performance and maintenance .....	41

## Abbreviations/Acronyms

Acronym/Abbreviation	Definition
API	Application Programming Interface
APP	Application
AI	Artificial Intelligence

# Chapter 1: Introduction

## 1.1 Background and scope of the project

The idea was conceived during the third year of study when the group was tasked to research on different topics and come up with a concept note. The group researched on the importance of application programming interfaces on the health sector. Application programming interfaces go a long way in improving service provision in health facilities across the country. I.e. Information can be available with the click of a button and on demand, hence greatly improving on the response time and emergency handling in hospitals.

### Scope:

The health API and mobile application aims at integrating multiple hospital databases across the country through the Application programming interface. The services provided by this api will then be consumed by our mobile application that provides features like a chatbot which will provide recommendations based on the communicated symptoms, Artificial Intelligence powered reminders for reminding patients to take the medicine, Data Explorer which provides a statistical graph and number for the people tested and this will help to statistically predict an outbreak of a disease prevalence, Prescmod which will be used by the health worker to enter details of prescription into a database, Medical Report which is for viewing and updating patient medical reports if necessary.

## Objectives

- To integrate databases through the Api thus enabling patient data be accessed from any hospital with the system through the medical reports module.
- To send timely reminders to the patient concerning their drug prescription.
- To recommend possible causes of symptoms experienced by the patient and a course of action to the patient through the chatbot.

## 1.2 Overview of the document

This document describes the implementation, testing and validation findings for the health API and mobile application system. It is divided into the following sections:

Section 1: This section gives an overview of the document

Section 2: System Specifications. This section describes and specifies the system completely and is the basis for the validation process.

Section 3: Design Output. This section describes the development tools used to implement the system and documentation provided as output from the design.

Section 4: Inspection and Testing. This describes the planning and documentation of the system test plan.

Section 5: Installation and System Acceptance Test. This section describes the validation of the installation process, which ensures that all system elements are properly installed in the host system

and that the user obtains a safe and complete installation, especially when installing software products.

Section 6: Performance, Servicing, Maintenance and Phase out. This section includes documentation of service and support concerning maintenance, future updates, problem solutions, requested modifications. It also describes the requirements for service, maintenance, performance, and support.

Section 7: Conclusions and Recommendations. This section describes a conclusion about the whole report.

## Chapter 2: System Specifications

The Health API and mobile application consists of the following features;

### a) **The mobile Application**

The mobile application has two user types I.e., the patients and the health workers.

### **HEALTH WORKERS**

The health workers' application has three modules:

#### **Data Explorer:**

The data explorer module uses data analytics to analyze patient data I.e., the patients tested for different diseases for our case we used a case study of few diseases which included .....When different patients are given prescription based on the tested disease the data for the disease is used to plot a graph and it used to determine which month has the highest disease prevailing and a solution is designed on how to reduce that and also that data is used to determine an outbreak of a disease in a particular region so as to generate graphs and guide health experts on a recommended course of action.

#### **Prescmod:**

Once a patient has been diagnosed and a recommended treatment given to the patient. The health worker then inputs this prescription for the case of tablets.

The prescmod module is linked to the notification module that sets off a reminder to the patients whenever the dose is due for taking.

### **Medical report:**

Once a patient presents themselves to the health facility, their medical report is searched for so that the doctors can get to know the patient history of the patients they are to work on through the medical reports viewed in this module. The health professionals can also update the patients' medical report based on their findings.

### **PATIENTS**

The patients' application has two modules:

#### **AI Powered Reminders:**

This module will provide reminders automatically with the health of generative AI to patients in case the health professionals give them prescriptions. These reminders will run according to the time of the prescription and they will be able to cancel themselves automatically in case they prescription is done.

#### **Doctor Smart:**

This module will provide a chatbot used for recommendations with the help of natural language processing to patients. Here patients will have to enter their symptoms and are given recommendations (we focused on malaria as a case study)

#### **b) The Health API**

It will be responsible for connecting to the different databases and fetching data to be used within the mobile applications.

## **2.1 Version of requirements and Version Control**

The requirements specified in the System Requirements Specification document (Version 1.0) were sufficient and approved. The team considered these requirements throughout the process of design and implementation of the system.

The Health API and mobile application system has one version, which is v1.0. We used Git and GitHub for version control during the mobile application development i.e., for both backend and front end. The source code for both is available and can be accessed on GitHub.

## 2.2 Input

The system receives input from the user.

**Input1:** User credentials, these include the email and password that users use when registering or logging into the mobile application. If someone enters the wrong email or password they are prompted to try again, if the user has forgotten their credentials there is an option for resetting their password where an email is sent to them for resetting.

**Input2:** Symptoms, The system allows users to enter their symptoms which are signs of illness that a user enters into the chatbot in order to get recommendations on what steps to take to try and combat them for example headache, high temperature etc. if the chatbot has no advice for a symptom, the user is prompted to contact a medical personnel

**Input3:** Possession based data, the system takes possession-based data like the name and health personnel enter it into the hospital databases when a patient visits a hospital for testing. The data can be accessed using the API to those who have permission to access it

**Input3:** Diagnostic test data, the system takes the kind of disease a patient has and health personnel enter it into the hospital databases when a patient visits a hospital for testing. The data can be accessed using the API to those who have permission to access it

**Input4:** Medical prescription, the system takes prescription given to a patient and health personnel enter it into the hospital databases when a patient visits a hospital for testing. The data can be accessed using the API to those who have permission to access it

**Input5:** Chemical Agent, the system receives the type of drug prescribed for a patient and it is entered into the hospital databases by health personnel when a patient visits a hospital.

## 2.3 Output

**Output1:** Analytics

Data format: graphs and numbers

Presentation: Graphs that show how the tested disease numbers are distributed across the different districts. The number of patients tested for a particular disease is plotted against the months of the year for a combined bar graph to show which disease is most and in which month. The table of

districts and numbers is also presented below the graph showing the number of people tested from each district and the number of people tested.

**Output2:** Recommendations

Data format: text

Presentation: the response generated by the chatbot after a user has asked it a question about their symptoms. The chatbot gives users the possible remedy for the symptom they are suffering from if it can't then prompts them to seek professional help.

**Output3:** Reminder is triggered if the time for taking the medication has reached as prescribed by the health professionals

## 2.4 Functionality

The Health API and Mobile Application system has the following functionalities:

- The system enables registration of health professionals and patients using the mobile applications.
- The system provides recommendations to patients through the communicated symptoms using the chatbot. Patients use the chatbot to enter their symptoms and they are given recommendations accordingly (we used malaria as a case study) through the patients application.
- The patients' mobile app sends AI powered reminders to the patients when it is time for taking their medication.
- The health workers app enables the health professionals to enter the name, drug, prescription, number of days and disease of the patient that has been tested and given prescription.
- The system provides a module to view the patient medical reports by the health professionals so that they can know the patient's medical history through the medical reports and also enable health professionals update the patient's medical reports
- The system provides a statistical graph that shows the number of diseases tested against the months of the year and number of people tested from a particular area. This helps to determine an outbreak of a disease from a particular area so that it can be focused on and try to control the outbreak.
- The system provides an api that will be used to connect to the different hospital databases and used to fetch and post data in the databases.



## 2.5 Limitations and safety

**Dependency on Active Internet Connection:** The system relies on an active Internet connection for proper functioning. In situations where there is no Internet connectivity, certain features may be limited or inaccessible.

**Precaution:** clear notifications and guidance to users when there is no internet connection, indicating which features may be limited or unavailable in offline mode. Additionally, implementing local caching of data where possible to allow users to access certain functionalities offline.

**User Error:** There is potential for user errors. Users may inadvertently enter invalid or inconsistent data, which could affect the accuracy or reliability of the system's outputs.

**Precaution - Precaution:** Implement comprehensive input validation mechanisms to detect and handle incorrect or inconsistent user inputs. Display helpful error messages

**Compatibility with Specific Devices or Platforms:** There may be limitations in terms of device or platform compatibility. It may work optimally on certain operating systems or require specific hardware configurations, potentially limiting its accessibility to a broader range of devices or users. For example, the application is an android one and cannot work on IOS

**Precaution:** Clearly communicating the supported devices, operating systems, and recommended configurations to users.

**Scalability and Performance:** As the user base or data volume increases, the system's performance and scalability may be tested. The system might face limitations in handling a large number of concurrent users or processing substantial data sets efficiently. Periodic performance monitoring and optimization can help address these limitations.

**Precaution:** Monitoring system performance regularly to identify potential bottlenecks or scalability limitations

## 2.6 Default settings

When the system is first installed, it has no accounts and the user is required to register with their email and password.

The password can be changed by pressing the reset password at the bottom of the login page.

For security reasons, the password is required to be at least 8 characters long.

**Notification Preferences-** By default, the system is set to send AI Powered reminders to patients in case they have a prescription. The notifications include information such as the type

Of the drug they have to take and time they are taking it.

## 2.7 Special requirements

Requirements the system is committed to;

- Our code is closed source.
- All data entered or output is kept confidential and sensitive data like passwords is encrypted for security.

- Confidentiality

Strict access controls to ensure that only authorized individuals have access to patient data and other confidential information.

Confidentiality agreements with participating hospitals and healthcare providers to protect patient privacy and prevent unauthorized disclosure.

Anonymization and de-identification of patient data when generating analytics and trends to protect individual privacy.

- Back-up of Records.

Regular and automated backup procedures to ensure the integrity and availability of data in case of system failures or data loss events.

Off-site storage of backups to prevent data loss due to physical damage or disasters at the primary site.

- Protection of Code and Data.

Code version control using tools like Git to track and manage changes, ensuring code integrity and facilitating collaboration among developers.

Secure code repositories with appropriate access controls to prevent unauthorized code modifications or disclosure.

Regular backups and redundancy measures for code repositories and data storage to mitigate risks of code or data loss.

### **Precautions**

Comprehensive input validation to prevent incorrect data entry and reduce the risk of errors or inconsistencies in the system.

Regular system maintenance and updates to address security vulnerabilities, performance issues, and user feedback.

User training and documentation to ensure correct usage of the system and proper handling of sensitive data.

## **2.8 Errors and alarms**

### **Data Inconsistencies or Inaccuracies:**

Error: Incomplete or incorrect data entries, duplication of records, or data entry errors.

Handling: Implementation of data validation mechanisms during data entry to ensure completeness and accuracy. Use techniques such as input validation, data verification checks, and real-time feedback to guide users in entering correct and consistent data. Perform regular data audits to identify and rectify any inconsistencies.

### **Security Breaches or Unauthorized Access:**

Error: Potential unauthorized access, data breaches, or hacking attempts on the system's databases or user information.

Handling: Implement robust security measures, including secure authentication and authorization mechanisms, encryption of sensitive data, and regular security assessments. Stay updated with security best practices and promptly address any identified vulnerabilities. Have an incident response plan in place to handle security breaches and data breaches promptly and effectively.

### **System Downtime or Performance Issues:**

Error: System crashes, prolonged downtime, slow response times, or inadequate system performance.

Handling: Regularly conduct system maintenance and updates to address performance-related concerns. Have a backup and disaster recovery plan in place to minimize downtime and ensure data availability in case of system failures.

**Compatibility Issues:**

Error: Compatibility issues with specific devices, operating systems, or web browsers, leading to limited functionality or incorrect rendering.

Handling: Conduct thorough compatibility testing across a range of devices, operating systems, and web browsers to identify and address any compatibility issues. Provide clear system requirements and recommendations to users regarding supported platforms and configurations. Regularly update and optimize the system to accommodate evolving technology trends.

**User Input Errors or Misuse:**

Error: Users entering incorrect or inconsistent data, misusing system functionalities, or misinterpreting system outputs.

Handling: Implement comprehensive input validation mechanisms to detect and handle user input errors. Provide clear instructions, tooltips, and user guidance to minimize input errors and promote correct usage. Offer contextual help and documentation within the system to educate users on the proper use of functionalities and interpretation of outputs. Conduct regular user training sessions to enhance user understanding and minimize misuse.

**Integration or Data Synchronization Issues:**

Error: Problems in integrating with hospital databases, inconsistencies in data synchronization, or failure to retrieve or update data properly.

Handling: Establish clear communication channels with hospital IT teams to address integration challenges. Perform thorough testing and validation of data synchronization processes. Implement error handling mechanisms and logging to identify and resolve integration issues promptly. Regularly monitor data synchronization and address any discrepancies in a timely manner.

## Chapter 3: Design output

### 3.1 Implementation (coding and compilation)

#### **Visual Studio Code:**

The application was developed using visual studio code integrated development environment that supports extensions, dependencies and tool kits for a variety of frame works including flutter.

#### **Flutter application development platform:**

We used flutter with dart programming language to build the health worker and patient mobile applications.

#### **Python programming language:**

We used python programming language in natural language processing to develop the doctor smart module. It is a chatbot that assists patients with signs and symptoms and gives a response based on how we trained it.

#### **Integration:**

##### **The API:**

**Spring framework:** we built the API in spring framework using java-programming language.

**Postman:** the API was tested using postman. Postman offers powerful testing capabilities.

**Apigee:** the API proxy was hosted in apigee, an API management tool on the google cloud platform

### 3.4 Documentation

A Software Design Document for the Health API and Mobile Application was written as output from the Design process. This document describes the architecture and system design and was intended for the project supervisor, system developers, system testers, patients and health professionals.

Table 1 Design Details

Topics	Design output	
<b>Good programming practice</b> <i>Efforts made to meet the recommendations for good programming practice...</i>	Source code is... <input checked="" type="checkbox"/> Modulized <input checked="" type="checkbox"/> Encapsulated <input checked="" type="checkbox"/> Functionally divided <input type="checkbox"/> Strictly compiled <input checked="" type="checkbox"/> Fail-safe (handling errors)	Source code contains... <input type="checkbox"/> Revision notes <input checked="" type="checkbox"/> Comments <input checked="" type="checkbox"/> Meaningfull names <input checked="" type="checkbox"/> Readable source code <input type="checkbox"/> Printable source code
<b>Dynamic testing</b> <i>Step-by-step testing made dynamically during the implementation...</i>	<input checked="" type="checkbox"/> All statements have been executed at least once <input checked="" type="checkbox"/> All functions have been executed at least once <input type="checkbox"/> All case segments have been executed at least once <input checked="" type="checkbox"/> All loops have been executed to their boundaries <input type="checkbox"/> Some parts were not subject to dynamic test Comments:	

# Chapter 4: Inspection and testing

## 4.1 Introduction

Table 2 Inspection and Performance

Topics	3.3.1 Inspection plan and performance	Date / Initials
<b>Design output</b> <i>Results from the Design Output section inspected...</i>	<input checked="" type="checkbox"/> Program coding structure and source code <input checked="" type="checkbox"/> Evidence of good programming practice <input checked="" type="checkbox"/> Design verification and documented reviews <input type="checkbox"/> Change-control reviews and reports Comments:	15th/06/2023 J.G.K, H.E, I.N, R.E.N
<b>Documentation</b> <i>Documentation inspected...</i>	<input checked="" type="checkbox"/> System documentation, flow charts, etc. <input type="checkbox"/> Test results <input checked="" type="checkbox"/> User manuals, On-line help, Notes, etc. <input checked="" type="checkbox"/> Contents of user manuals approved Comments:	16th / 06 / 2023 J.G.K, H.E, I.N, R.E.N
<b>Software development environment</b> <i>Environment elements inspected...</i>	<input checked="" type="checkbox"/> Data integrity <input checked="" type="checkbox"/> File storage <input checked="" type="checkbox"/> Access rights <input type="checkbox"/> Code protection <input type="checkbox"/> Installation kit, replication and distribution Comments:	17th/06/2023 J.G.K, H.E, I.N, R.E.N

<i>Topics</i>	<b>3.3.1 Inspection plan and performance</b>	<i>Date / Initials</i>
<b>Result of inspection</b> <i>Approval of inspection.</i>	<input checked="" type="checkbox"/> Inspection approved Comments:	18th/06/2023 J.G.K, H.E, I.N, R.E.N

## 4.2 Test plan and performance

*Table 3 Test Plan*

Item to test	How to test	Expected output
API	Api to be tested with postman	Api endpoints should checkout
medical reports module	inputing a given name into the search field (a name must have been pre-registered ) and observing the outcome  trying to filter search results by year to narrow the search results	medical records relating to that name should be returned  when search filter is used , records example : 2019, records for 2019 should be returned
doctor smart module	by making simple requests like asking for solutions to headache or inputting symptoms relating to a disease	the chatbot is expected to recommend over the counter treatment like aspirins, Panadol or to see a doctor



### 4.2.1 Test objectives

The main objective is to verify that the functionality of the health Api and mobile application system works according to the specifications provided in the software design document, and the requirements specification.

Other objectives include:

To communicate to the responsible parties the items to be tested the schedule, the test budget and defined environmental needs for testing.

To improve on the quality of the software

To identify and locate any errors in the health API and mobile application.

*Table 4. Test Objectives*

what was tested	how it was tested	why it was tested
overall functionality of the Health-API and mobile application	The system was tested by simulating different hospital management systems with different databases i.e. Mysql, postgresql, that had been linked through the Api and containing dummy data-API calls were then made.  we also used postman to test the API	To ensure that all modules of the Health- API and mobile application work as required.
systems compatibility across platforms	the application was installed on both android and iOS and then run	to ensure that the application runs smoothly on a range of devices and can be run on multiple devices



### **4.2.2 Scope and Relevance of tests**

The scope of the tests carried out were based on modular-based coverage

Module-based coverage is based on the modules that are the doctor smart module, data explorer module, prescmod module, notifications and finally the Application-programming interface.

The tests were designed to cover for both verification and validation testing.

Verification covered the systems requirements during the requirements phase. Validation covered testing during the implementation phase.

Testing also, helps ensure the accuracy of the requirements vis a vis the actual product, system performance under heavy load and stress.

Table 5. Scope and Relevance

System Requirements	Test Coverage (%)	Test Objective met
the system should provide a combination of health records from various institutions	100	the system provides a combination of health records
The system should provide a search criterion from which health institutions can search for data to view and update if necessary at a given instance i.e., from a given timeline	100	filtering feature is functional
the system should send notifications to patients undergoing treatment through oral administration to take their prescribed drugs by a doctor	100	patients are able to receive and view notifications on their devices
the system should display a graphical and tabular representation of diseases tested by area and number of cases	100	Analytical data is displayed via the data Explorer module
the system should allow the user input signs and symptoms through the Doctor Smart (which is a chatbot feature)	100	The chatbot feature is functional. a user is able to interface with this feature
the system should allow health professionals enter patient details after testing	100	The prescmod feature is functional a user able to interface with it.

### **4.2.3 Levels of tests**

#### **Module test.**

We developed the application in modules therefore the need to perform modular tests. Each of the application modules was tested independently.

#### **System Acceptance Tests.**

To ensure the system complies with the system requirements specification.

#### **Unit Testing.**

We tested each module individually before we could integrate and test. Below are some of the modules we tested;

##### **Doctor Smart module.**

The module was tested to ensure that appropriate responses were given to the user. The trained chatbot returned the appropriate response to the user.

##### **Reminder.**

The reminder module was tested to ensure that users receive notifications for taking the prescribed medicine at the right time. The right notifications were displayed and a list of notifications could be viewed under the Reminder module.

**API.** We used postman to test the API. This was to ensure that the data could be fetched from various databases. The test coverage was 100%.

##### **Medical Report.**

The medical report module was tested to ensure that the system could return medical records as requested. The system was verified to do so.

##### **Integration test.**

After successful testing of individual modules. Integration testing was performed to ensure

## 4.2.4 Types of tests

**User acceptance tests.** We carried out user acceptance tests by sampling a number of android and iOS devices. This was to ensure ease of use of the system and a great user experience.

### **Input Tests.**

We carried out input tests by generating dummy data from different databases we created; we also input data into the input fields of various modules like the doctor's prescmod, the doctor smart module.

**Functionality tests.** Functional tests were performed vis a vis the functional specifications. Functional tests examined the corresponding output for a given set of inputs into the system.

## 4.2.5 Sequence of tests

Table 6. Test Case summary

<i>TEST ID</i>	Test case description	Test procedures	Expected results	<i>comments</i>
001 Registration	Check if users can be registered successfully.	input details as required for each user role	Popup of user successfully registered. user can then proceed to sign in page	user was registered
	Verify that the the patient can log into the patient application	login using registered details	The patient is taken to the patient home page on successful sign in.	test case passed
	Verify that the health worker can login the health application	login using registered details	a health worker is taken to the health worker application home page on successful sign in.	health worker signed in successfully
002	<b>MEDICAL REPORTS MODULE</b>			
	Verify that the doctor can search for medical records by name.	in the search bar, input a patient name or full names	a list of records synonymous with this name are returned	passed
	Verify that the doctor can filter search results by year	after inputting patient details , click the filter icon and pick a year	medical records for that particular name are returned for the respective year	passed
	Verify that the doctor can view medical records	click on any medical record from the list of	the records should be able to open and viewed	passed

		returned records		
003	<b>Doctor Smart module</b>			
	Verify that the doctor smart button links to the chatbot page	In the patient application, once signed in click the Doctor smart button.	The button should lead the user to the chat screen and a welcome message displayed.	Test case passed. a patient was taken to the chat screen as expected
	Verify that a user can type text into the provided input field	try to input text and wait for a response	an appropriate response or error message must be displayed	passed
	Verify that the response is returned within 2 seconds of sending	try to input text and wait for a response	an appropriate response or error message must be displayed within the specified timeframe	passed
003	<b>Reminder module</b>			
	Verify that the reminder button works	click on the reminder button	user must be led to notifications page	passed
	Verify that a list of notifications can be displayed	click on the reminder button	user must be able to see notification reminders	passed
004	<b>Data Explorer module</b>			
	Verify if the graph is plotted according to the disease tested against months of the year			<i>passed</i>
005	<b>Prescmod Module</b>			



	Verify that all form fields are available	click on the prescmod button	the health worker must be able to view a prescription form	form is provided
	Verify that a form input can be saved	fill in the form fields and click update	an update message should be displayed	passed

### **4.2.6 Configuration and calculation tests**

- A patient (user) enters their symptoms and communicates with the chatbot to get recommendations regarding their symptoms.
- A health professional enters the details in the prescmod module, which trigger the AI powered reminders.
- A health professional hits an API endpoint with a request of the patient's name, which fetches them the patient's medical report.

## **4.3 Precautions**

### **4.3.1 Anomalous conditions**

Anomalies are events that differ from the standard events defined in the application.

Therefore, anomalous conditions in our application refer to the deviations of system

From the user's expected behavior. From this application, the following are the possible anomalous Conditions that may occur during operation of the application.

The following are the anomalous conditions that arise from the system in operation:

- Inappropriate responses received from the doctor smart module. This arises from failure to interpret some user queries or inputs
- Setting a false reminder. This arises from not updating the database with the number of days for the prescription.
- Wrong analytical visual data. This may arise from the health professional forgetting to enter the patient tested.

### **4.3.2 Precautionary steps taken**

# Chapter 5: Installation and system acceptance test

## 5.1 Input files

*Table 7 Input Files*

File	Use
main. Art	The main entry point of the Flutter application.
pubspec.yaml	Specifies the project dependencies and configuration.
lib/	Directory containing the main app logic and code files.
- Main. Art	Contains the main code for initializing the app.
- chatbot.dart	Implements the chatbot functionality using Python code.
- Authentication. Art	Handles user authentication using Firebase.
- screens/	Directory containing the various app screens.
-- home_screen.dart	The home screen of the app, displaying relevant information.
-- chat_screen.dart	The screen for interacting with the chatbot.
assets/	Directory for storing static assets used in the app.
- images/	Contains image files used in the app's UI.
- fonts/	Stores custom font files used in the app's UI.
android/	Directory specific to the Android platform.
- app/	Contains the Android app configuration and resources.
- ...	Other Android-specific files and directories.

## 5.2 Supplementary files

Table 8. Supplementary files

File	Purpose
README.md	- Provides an overview of the app and its key features. Guides users through setup and installation steps.
chatbot/README.md	- Provides details about the Python-based chatbot functionality. Offers instructions for setup and usage.
firebase/README.md	- Provides information about Firebase integration and user authentication. Guides setup and configuration.
contributing/README.md	- Provides guidelines for contributing to the project. Explains the contribution process and standards.
license/README.md	- Contains information about the app's licensing. Specifies the open-source license under which it is released.

## 5.3 Installation qualification

**Table no: Checklist of the Installation and system acceptance test**

Table 9. Installation summary

Topics	Installation summary
<b>Installation method</b>	<input checked="" type="checkbox"/> Automatic - installation kit located on the installation media <input type="checkbox"/> Manual - Copy & Paste from the installation media Comments:
<b>Installation media</b>	<input type="checkbox"/> Diskette(s) <input type="checkbox"/> CD-ROM <input type="checkbox"/> Source disk folder (PC or network) <input checked="" type="checkbox"/> Download from the Internet Comments:

<i>Topics</i>	<b>Installation summary</b>
<b>Installed files</b>	<ul style="list-style-type: none"> <li>• APK file</li> <li>• App icon</li> <li>• Splash screen image or animation</li> <li>• Custom font files</li> </ul>

**Table no: Installation Procedure Check**

*Table 10. Procedural check*

<i>Topics</i>	<b>Installation procedure</b>	<i>Date / Initials</i>
<b>Authorization</b>	Person responsible: Noel	23/07/2023
<b>Installation test</b>	<input checked="" type="checkbox"/> Tested and approved in a test environment <input type="checkbox"/> Tested and approved in actual environment <input type="checkbox"/> Completely tested according to test plan <input checked="" type="checkbox"/> Partly tested (known extent of update) Comments:	19/07/2023

# Chapter 6: Performance, servicing, maintenance, and phase out

## 6.1 Service and maintenance

As software systems evolve, it is crucial to address various factors to maintain the robustness and reliability of the system. These factors include changes in the operating environment, technological advancements, updates to programming languages, and other relevant considerations. To ensure the continued effectiveness of the system, regular servicing and maintenance are necessary.

The development team responsible for this particular Health API and Mobile Application System remains dedicated to working closely with stakeholders to address any anomalies that may arise during system usage. The current version of the system is just the beginning, as it will continue to evolve based on user feedback and concerns. The service and support documentation for this system is designed to provide comprehensive information on maintenance, future updates, problem solutions, and requested modifications. It serves as a valuable resource for both users and system administrators, enabling them to optimize the systems performance and resolve any issues that may occur. The documentation covers the following key aspects;

**User Manual.** A comprehensive user manual has been created to guide users through the functionalities and features of the system. It provides step-by-step instructions on how to navigate the mobile application, interact with the API, utilize the chatbot, interpret analytics graphs, view patient medical history through reports and also enter their information for prescription. The user manual aims to empower users to effectively utilize the system and find solutions to their queries.

**Support Channels.** The documentation tells users where to go for help. It provides contact information for customer support, such as a phone number or email address. Users can reach out to these support channels to get assistance when they have questions or face difficulties.

**Troubleshooting Guide.** The troubleshooting guide helps users fix common problems they may encounter the system. It provides step-by-step instructions on how to identify issues and offers solutions or workarounds. Users can use this guide to solve problems on their own.

**Maintenance Guide:** The maintenance guide documents the necessary procedures for system maintenance. It includes instructions on regular backups, database maintenance, and monitoring the health of the system. The guide also outlines troubleshooting techniques to diagnose and resolve common issues that may arise during system operation. It ensures that any potential downtime or disruptions are minimized through proactive maintenance measures.

**Future Updates Documentation.** As part of the system's development roadmap, future updates and enhancements are anticipated. Documentation on future updates outlines the planned features, improvements, and bug fixes. It includes a release schedule, versioning information, and any prerequisites or considerations for applying updates. This documentation enables the development team to streamline the update process and ensure smooth transitions for users.

**Modification Request Procedures.** In order to address requested modifications and feature enhancements, a documented procedure is in place. Users can submit modification requests through designated channels, including a dedicated support email or online form. The procedure outlines the process of capturing, evaluating, prioritizing, and implementing user requests. This ensures that user feedback and enhancement suggestions are properly documented and considered for future updates.

## 6.2 Performance and Maintenance

Many factors will inform the need for maintenance, these include too much delay in the synchronization of data to the online database for example beyond one minute. The other factor is where software component upgrades are required for example migrating to newer version of the programming language or the need to adapt a new technology on the market.

The team will be able to provide the following support where need be.

Training of users on how to use the system. There will be training sessions on the site of the users to enable users learn how to use the new system.

Installation, integration and configuration of the system. The time will do the installation and any necessary integrations.

Software upgrades. Software upgrades will be informed by new user requirement and bugs discovered. The software development lifecycle will be followed from start to end and all the necessary artifacts will be produced during upgrades.

Debugging in case of any bugs discovered. Bugs that require immediate attention and rectifying will be documented post implementation.

**Service Requirements.** The system is expected to be available and accessible to users without significant interruptions. The service should have a high uptime percentage, aiming for at least 99% availability. It should be capable of handling a substantial number of concurrent users and requests, ensuring scalability to accommodate future growth. The system should also support secure and encrypted communication channels to protect sensitive user data.

**Maintenance Requirements.** Regular maintenance activities are crucial to keep the system running smoothly. This includes periodic backups of data to ensure data integrity and availability.

Maintenance tasks should be performed without disrupting the system's normal operation.

Documentation for maintenance procedures, as mentioned earlier, should be available to guide the maintenance team and ensure consistency.

**Performance Requirements.** The system should deliver optimal performance to ensure prompt responses and minimize user wait times. Maximum time taken before delivering output (response time) should be within acceptable limits to provide a seamless user experience. Performance monitoring and logging mechanisms should be in place to identify bottlenecks, optimize system resources, and proactively address performance issues.

**Support Requirements.** Clients expect various levels of support for the system. This includes providing assistance for user inquiries, troubleshooting technical issues, and resolving problems encountered during system usage. Support channels, such as email, a dedicated support portal, or a helpdesk system, should be established to receive and address user concerns. The support team should be knowledgeable about the system and equipped to provide timely and effective assistance.

**Incorporating Changes.** Changes to the system may be necessary to enhance functionality, address issues, or accommodate evolving requirements. Decisions regarding changes are made based on careful evaluation and analysis of user feedback, system performance data, and emerging industry standards. Changes should be prioritized, considering their impact on system stability, compatibility, and user benefits.



Software Upgrades. Upgrades are planned to ensure the system remains up to date with the latest technologies, security patches, and feature enhancements. Upgrades follow a structured process, including thorough testing, staging environments, and rollback plans, to minimize disruption during the upgrade process. Detailed documentation and guidelines are prepared to facilitate a smooth upgrade experience for users.

Revalidation. Whenever significant changes or upgrades are made to the system, a revalidation process is carried out to ensure the system continues to meet the required standards and functionality. The revalidation process involves comprehensive testing of the modified system to confirm that all critical functionalities operate as expected. Revalidation protocols and test cases are established to guide the process.

Phase-out and Data Migration. When transitioning from the old system to the new one, careful planning and data migration procedures are necessary. The migration process involves extracting data from the old system, transforming it to fit the new system's structure, and loading it into the new system. Detailed migration plans, including data mapping, validation, and verification steps, are prepared to ensure data integrity and accuracy during the transition.

**Table no: Performance and maintenance details**

*Table 11. Performance and maintenance*

<i>Topics</i>	<b>Performance and maintenance</b>	<i>Date / Initials</i>
<b>Problem / solution</b>	<i>Detection of system problems causing operating troubles. A first step could be to suggest or set up a well-documented temporary solution or workaround.</i>	Dates must be filled in

<i>Topics</i>	<b>Performance and maintenance</b>	<i>Date / Initials</i>
<b>Functional maintenance</b>	The API depends on data from different hospital databases and that data is constantly changing. the system will continuously be adjusted to ensure data consistency from the different databases	
<b>Functional expansion and performance improvement</b>	<ul style="list-style-type: none"> <li>• Conducting usability testing sessions with real users to gather feedback on the system's performance and identify areas of improvement.</li> <li>• Continuously releasing updates and bug fixes to address any reported issues, enhance system performance, and introduce new features.</li> </ul>	

# Chapter 7: Conclusion and Recommendations

In conclusion, the development of the health API and mobile application has been a significant achievement, providing valuable features and functionality to users in the healthcare domain. Throughout the project, we have focused on meeting the system requirements, ensuring security, confidentiality and quick access to health related solutions

The system offers users a convenient and efficient way to access health-related information, interact with a chatbot for automated symptom answers, and visualize patient trends through the analytics module. The implementation of Flutter for the mobile application, Python for the chatbot and Firebase for authentication has provided a robust and scalable foundation for the system. During the development process; we have taken several precautions to prevent wrong data and malfunction due to incorrect input or use, and have incorporated mechanisms for error handling and exception management. Additionally, we have recognized the importance of documentation, both for internal development purposes and for providing support to users and maintenance personnel.

Overall, the health API and mobile application system demonstrate our dedication to delivering innovative solutions in the healthcare sector. By leveraging the power of technology, we aim to improve access to information, streamline processes, and contribute to better healthcare outcomes. We are confident that the system will positively impact users' lives and set a benchmark for future developments in the field.

## Recommendations

Establishing a feedback mechanism to actively engage with users and gather their insights on the system's usability, features, and any areas of improvement. Regularly incorporating user feedback will help identify pain points, prioritize enhancements, and ensure that the system meets user expectations.

Enhancing Chatbot Capabilities. Invest in the further development and training of the chatbot to improve its accuracy, response quality, and ability to handle a wider range of user queries. Incorporate natural language processing (NLP) techniques and machine learning algorithms to enable the chatbot to understand user intent better and provide more accurate and personalized responses.

Enhancing Security Measures. Strengthening the system's security measures by implementing additional layers of encryption, authentication mechanisms, and access controls. Regularly perform security audits to identify and address any vulnerabilities and stay up-to-date with the latest security best practices.

## Appendix A: User Manual

A step-by-step manual that will guide you on how to use the Health API and Mobile Application

### Main Features:

The Health API and Mobile Application consists of the following main sections;

- a) **Register.** Gives the user the provision to register for the use of the mobile application.
- b) **Login.** Allows the user to access the system after registering for the system.
- c) **Doctor Smart.** This interface will allow the registered user to be able to get some recommendations according to the communicated symptoms.
- d) **AI Powered reminders.** Displays notifications on the prescriptions given to the patient.
- e) **Data Explorer.** This provides a graph for analyzing the tested patients and helps to predict an outbreak of a disease.
- f) **Prescmod.** It is used to enter the name, drug, prescription and number of days for the prescription given.
- g) **Medical Reports.** It is used to view the patient's medical history and also update patient medical reports.

### **How to get the application and Installation**

The application only runs on android phones. You will be provided with an APK to install on the mobile phone. When you attempt to install the application, you'll receive a prompt that you're installing an application from an unknown source. Head to settings and in security, enable installation from unknown sources. The application will then be installed successfully.

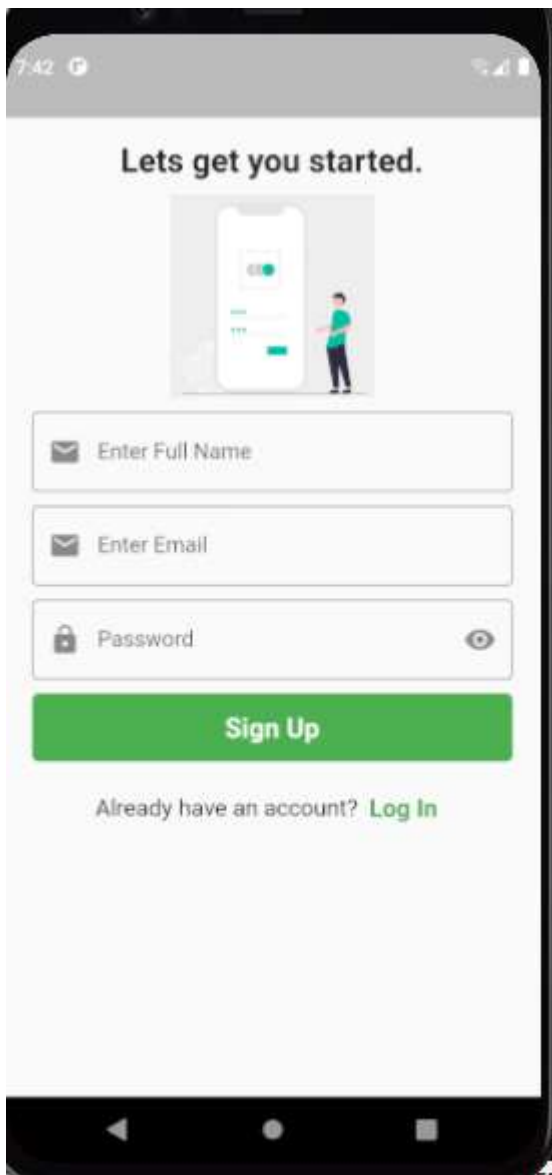
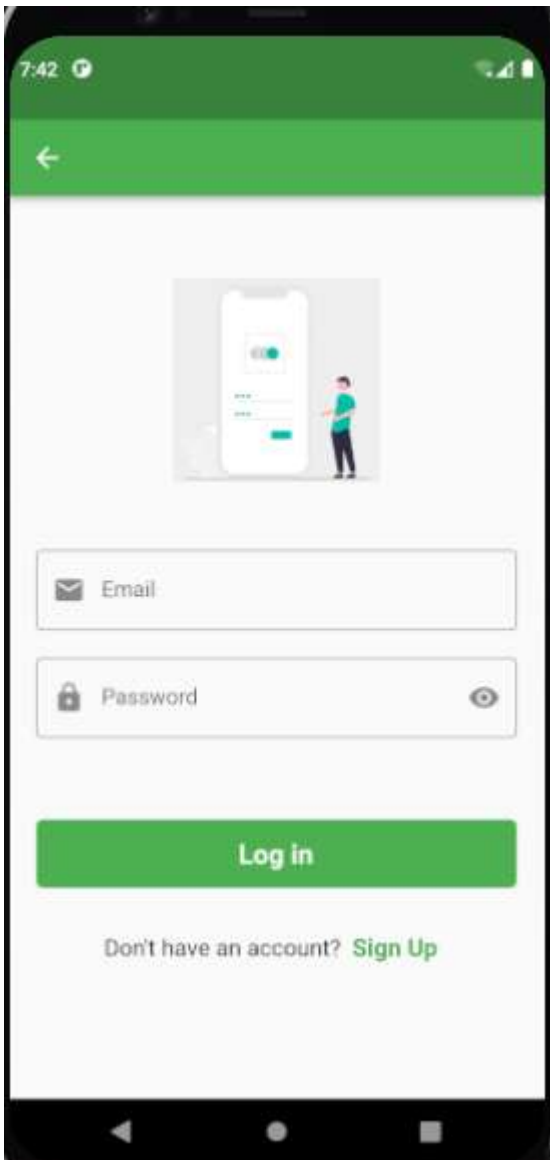


Figure 1. User registration

This screen is what the user sees when he first loads the application.

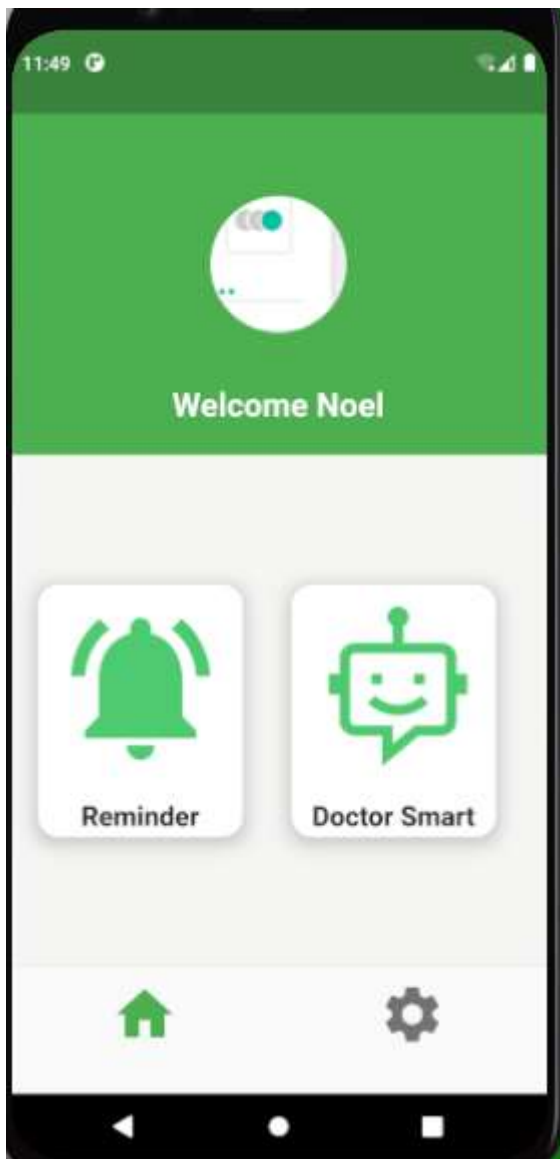
- If the user has no account, then they must provide credentials i.e., the full name, email and password which they provided to create the account.
- The SignUp button will enable the user to create an account.
- If the email does not follow the standard email procedure i.e. [example@gmail.com](mailto:example@gmail.com) , then the user will not be able to create an account.
- The slashed eye icon can be used by a user to display their password or hide it.



*Figure 2 user login*

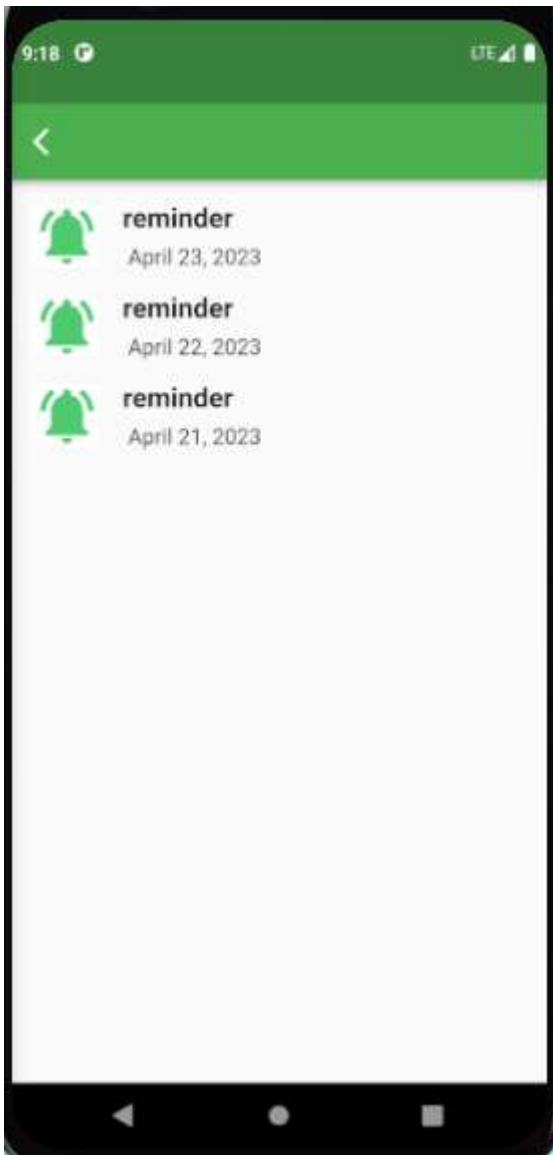
This screen is what the user sees when he first loads the application.

- If the user has an account already, then they must provide credentials i.e., the username and password which they provided while registering or creating the account.
- The login button will enable the user to log into the application.
- if the email does not follow the standard email procedure i.e. [example@gmail.com](mailto:example@gmail.com) , then the user won't be able to create an account.
- if the username or password is incorrect, then the user will not be able to log in.
- If the user is new to the application, then they click the create account text and they will be led to an interface where they will create an account enabling them to log in.
- The slashed eye icon can be used by a user to display their password or hide it.



*Figure 3. Patient home screen*

This screen is a landing page for the activities that will be provided to the user .i.e. doctor smart, which has a chatbot and reminder with AI powered reminders.



*Figure 4. Notifications*

This interface has the reminders for patient. They can be cancelled automatically when the prescription for the patient is done.



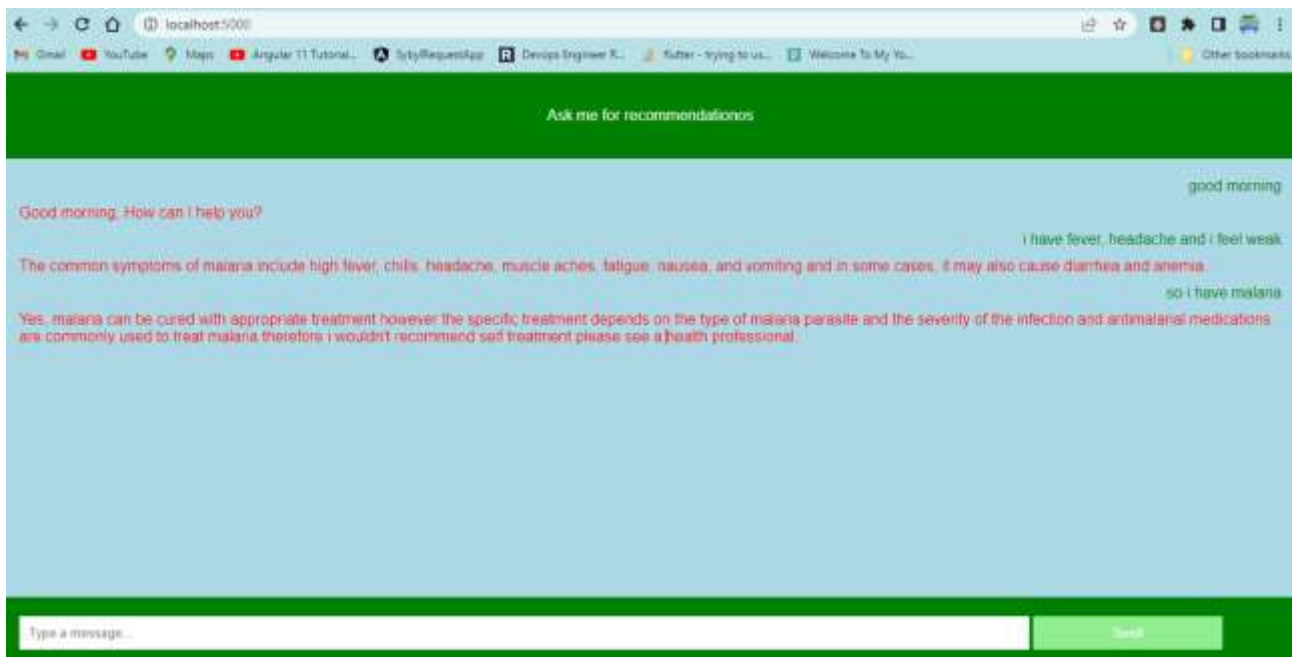
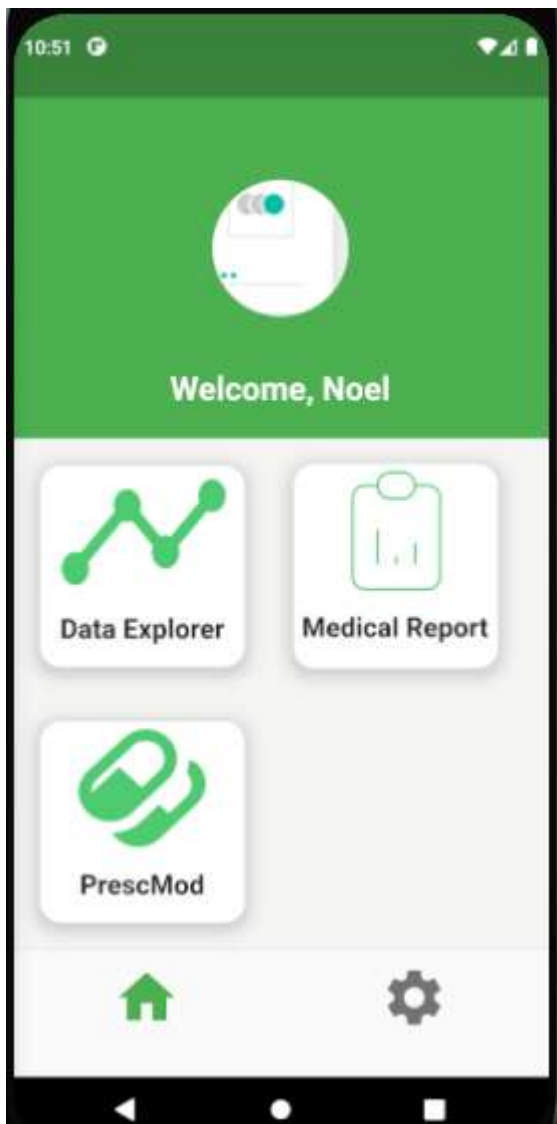


Figure 5. Chat screen

This interface is a chatbot that allows patients to get recommendations for their communicated symptoms.



*Figure 6. Health worker chatscreen*

This interface provides options to the health professional, which include data explorer, medical report and prescmod to interact with the system.

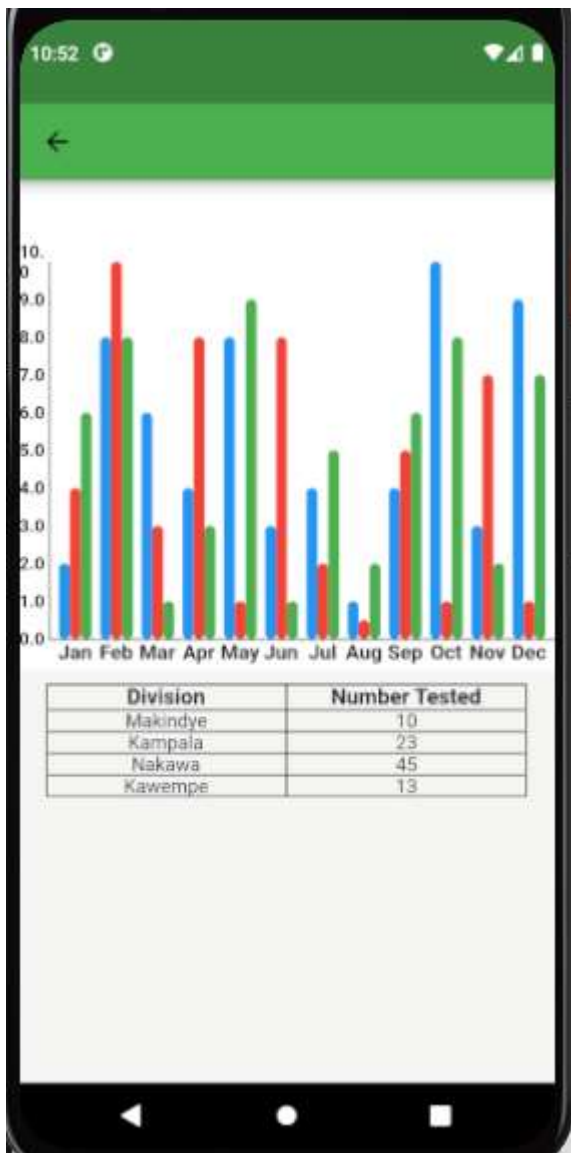
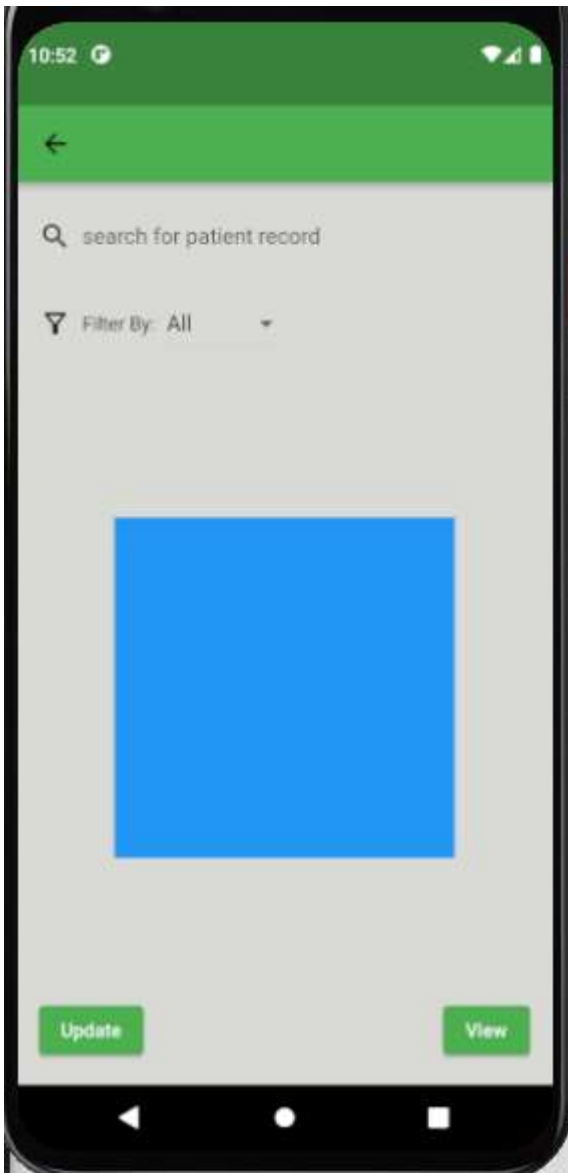


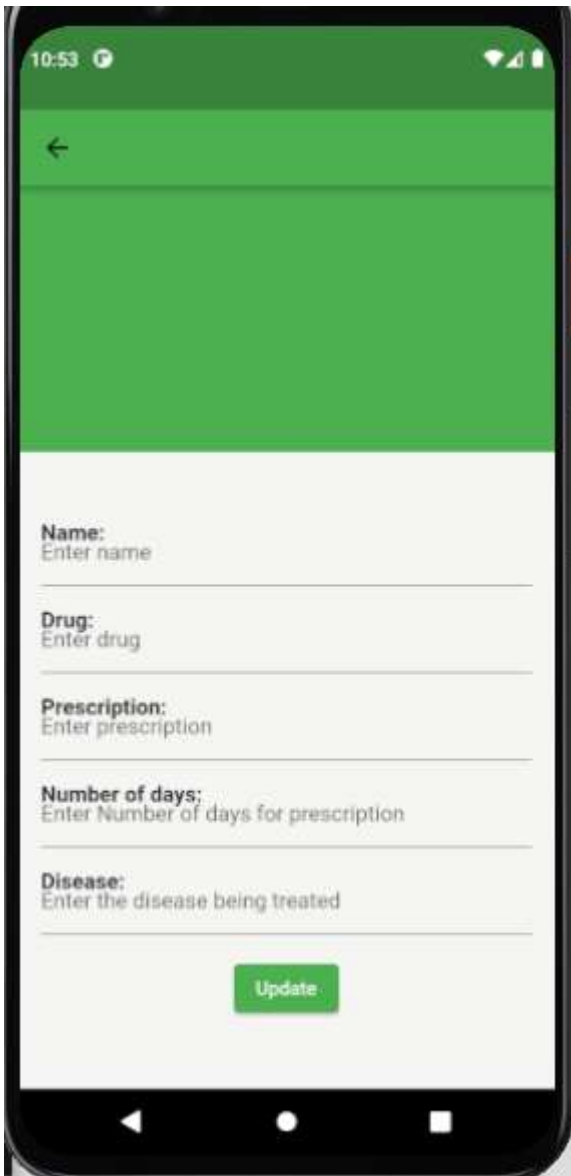
Figure 7. Data Explorer Home screen

This interface provides a graph for analytics to predict an outbreak in a certain area and try to control it.



*Figure 8. Medical records*

This interface provides a module for viewing past patient medical history and reports and helps them to updating the patient medical reports.



*Figure 9. Prescmod screen*

This interface is used by the health professional to enter the name, drug, and prescription, number of days and disease of the tested patient.

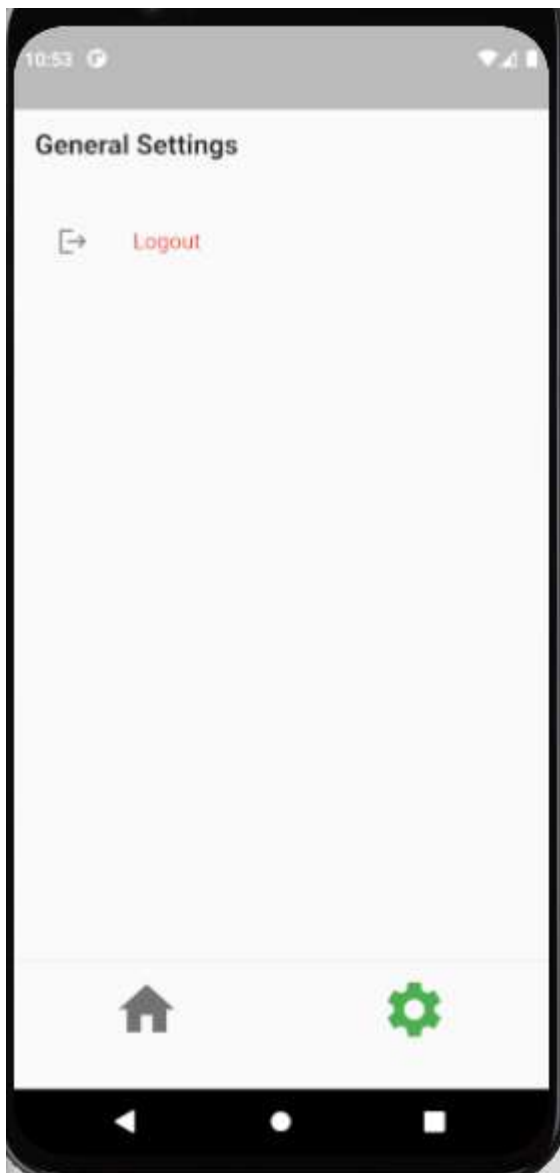


Figure 10. General settings

This interface is used to log out of the system.

Final approval for use	
Identification:	
Responsible for validation:	
Remarks:	
Date:	Signature:

