UiA University of Agder

eHealth

A smartphone-based e-Health application for enhancing rural healthcare with the integration of medical sensor devices.

PANKAJ KHATIWADA

SUPERVISOR

Associate Prof. Jan Pettersen Nytun
Associate Prof. Martin Wulf Gerdes
Associate Prof. Santiago Gil Martinez

University of Agder, 2019
Faculty of Engineering and Science
Department of ICT

UNIVERSITY OF AGDER

E-Health

A smartphone-based e-health application for enhancing rural healthcare with the integration of medical sensor devices

BY
PANKAJ KHATIWADA

Supervisor
Associate Prof. Jan Pettersen Nytun
Associate Prof. Martin Wulf Gerdes
Associate Prof. Santiago Gil Martinez

Masters Thesis
IKT 590

University of Agder, 24 May 2019
Faculty of Engineering and Science
Department of Information and Communication Technology

Abstract

The purpose of this research is to develop a smartphone or tablet based eHealth application to assist health workers in remote regions of different parts of the world with recording medical information and with the provision of basic health services to the patients. Health applications are becoming more popular day by day, and the use of technology along with these applications helps to improve the healthcare system. In this project, "mTeleHealth-UiA", a smartphone-based application, was developed to address this challenge. The application was implemented for the Android platform in the Android studio development environment, using XML and Java programming language. The app was designed to meet the identified requirements. It allows to create a health worker profile, under which new patient information can be collected and stored, the patients' vital signs can be measured with medical sensor devices, a score based on the vital parameters can be calculated to give recommendations for further follow-up, and the medical history can be checked to analyze the symptoms and support the diagnosis. Due to time limitations, we could not test this application in a real world scenario with real health workers and patients. Instead, after the implementation of the project, we tested the application with some test participants, using a questionnaire to obtain feedback based on their experience. The test result was analyzed, and most of the participants found it to be user-friendly and useful. Overall, they were satisfied with the process of information collection and suggestions provided to the patient. Feedback and proposals from the participants are essential and will be adopted in future work, and some are also proposed as part of further development in the near future.

Preface

This thesis work fulfills the requirement of my masters degree in Information and Communication Technology (ICT) at the University of Agder (UiA) in Grimstad, Norway. This thesis was performed during the spring semester 2019 IKT 590 at the Faculty of Engineering and Science with the Department of Information and Communication Technology (ICT).

My thesis project began with understanding the scenario from the lacking of basic health service by people in remote areas of different countries due to lack of expert medical people. Then, the research problem was created based on this scenario. The question was "How can a smartphone-based eHealth app help the rural area patients to get basic health service by the help of health workers?" The objective of this thesis was to develop the application for handheld devices such as smartphone or tablets and its main task is to support the health worker in the rural areas or remote areas of different countries to provide basic health support for the patient there.

I want to thank my supervisor, Associate Professor Jan Pettersen Nytun, Associate Professor Martin Wulf Gerdes and Associate Professor Santiago Gil Martinez who made my project possible through their essential feedbacks, suggestions, and guidelines. I also like especially thank Martin Wulf Gerdes for providing me with various medical devices and tablet which is necessary for the development of this application. Also, I would like to thank all the participants involved in the testing of the application.

Finally, I would like to thank my family and friends for their moral support and endlessly believing in me.

Production note: We use Latex as the tool for writing this thesis. Development is done in Android Studio while programming language used Java and XML.

Pankaj Khatiwada

24th May 2019

University of Agder, Grimstad, Norway

Contents

List of Tables

List of Figures

List of Abbreviation

A2DP	Advanced Audio Distribution Profile
AOSP	Android Open Source Project
API	Application Programming Interface
ATT	Attribute Profile
BLE	Bluetooth Low Energy
BR	Blood Pressure
BR/EDR	Bluetooth Basic Rate/ Enhanced Data Rate
CDA	Clinical Document Architecture
CDG	Continua Design Guidelines
CPU	Central Processing Unit
DEX	Dalvik Executable
DVM	Dalvik Virtual Machine
EHR	Electronic Health Record
EMR	Electronic Medical Record
ER	Entity Relational
EWS	Early Warning Score
FHIR	Fast Healthcare Interoperability Resources
GATT	Generic Attribute Profile
HAL	Hardware Abstraction Layer
HDP	Health Device Profile
HFS	Health & Fitness Server
HIS	Health Information System
HL7	Health Level-7
HTML	Hypertext Markup Language
ICT	Information and Communications Technology
JVM	Java Virtual Machine
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
IOT	Internet Of Things
ISO	International Organization for Standardization
ITU	International Telecommunication Union
JSON	JavaScript Object Notation

JVM	Java Virtual Machine
MEWS	Modified Early Warning Score
NEWS	National Early Warning Score
NFC	Near-Field Communication
OHA	Open Handset Alliance
OS	Operating System
PCHA	Personal Connected Health Alliance
PDA	Personal Digital Assistant
PHD	Personal Health Device
PHG	Personal Health Gateway
R2	Reaffirmation
RFCOMM	Radio Frequency Communication
SIG	Special Interest Group
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
UA	Upper Arm
UI	User Interface
USB	Universal Serial Bus
UUID	Universally Unique Identifier
VM	Virtual Machine
XML	eXtensible Markup Language
Wi-Fi	Wireless Fidelity

Chapter 1

Introduction

1.1 Motivation

According to the World Bank and the World Health Organization (WHO), half of the world population of around 3.8 billion people are still suffering from missing essential health services, and around 400 million people even lack basic health services. Furthermore, each year many people are the victim of poverty because they have to pay for health services from their own pocket [10]. This fact is completely unacceptable, and solutions should be found that allow all people on earth to be provided with basic healthcare services, when and where they need them, without facing any hardship. Almost all people being affected by the lack of such basic healthcare services reside mainly in rural areas of underdeveloped countries or developing countries. The main reason for this is a glut of health workers and medical experts in urban areas, and their aversion to move to rural areas and provide health services there. As a result, patients from rural areas have to travel long distances and spend much of their time and money to get basic health service. Hence, this challenge should be of high concern and should be addressed with the help of latest Information and Communication Technologies (ICT). The emerging capabilities of ICT plays a vital role in the field of Health and Medicine and is being the center of many research projects nowadays. Their importance for the healthcare system has been increased significantly, because it can lead to better quality, safety, and efficiency of the healthcare services [12]. Therefore, this problem is addressed by the eHealth application for mobile phones or tablet PCs (also known as Mobile Health or mHealth applications) to support the health workers in rural areas to collect the data from their patients such as patient information, vital life signs readings, family history, and medical history. Also, the application supports them with the symptom assessment and with an overview of the patients' medical status and to treat them accordingly.

The majority of health care providers prefer their first choice to serve in urban areas. This urban/rural is consistent whole over the world in every country which causes a barrier and significant challenge to provide health services. The reason behind this is, that these countries already suffer from acute shortages of health workers in the rural area. Also, vast number of the people, of these developing countries lives in a rural area compared to developed countries. These people living in the rural area are comparable poor than the people living in urban area.[33] The lack of health workers in their area causes them to travel for just a primary health service which can result in the expense of more money in their treatment and travel. Time is also another constraint which is being spent during complete treatment and travel. Even when they do travel to access health care, they are facing long queues and waiting times to receive simple or even poor quality of care in government facilities that provide low-cost care. This causes a delay of the diagnosis of the patients who might be affected by easily preventable or manageable illnesses, which can in turn result in a critical or worse health condition, and in some cases even in the death of patients.

The access to primary healthcare services is a basic human right, and the utilization of latest technologies can play an essential role in implementing this right. With the rapid growth of information technology, applications have been developed to assist front-line healthcare workers in rural areas. Telehealth and mHealth services provide the opportunity to connect doctors in urban areas with patients in rural areas in order to to provide healthcare, and by that to fill the crucial gap to healthcare services by providing access to healthcare through remote and on-site diagnosis, treatment and reducing time and distance travelled for basic health care. According to David H. Peters [30] access to healthcare services is mainly dependent on the four dimensions shown in figure 1.1 below.

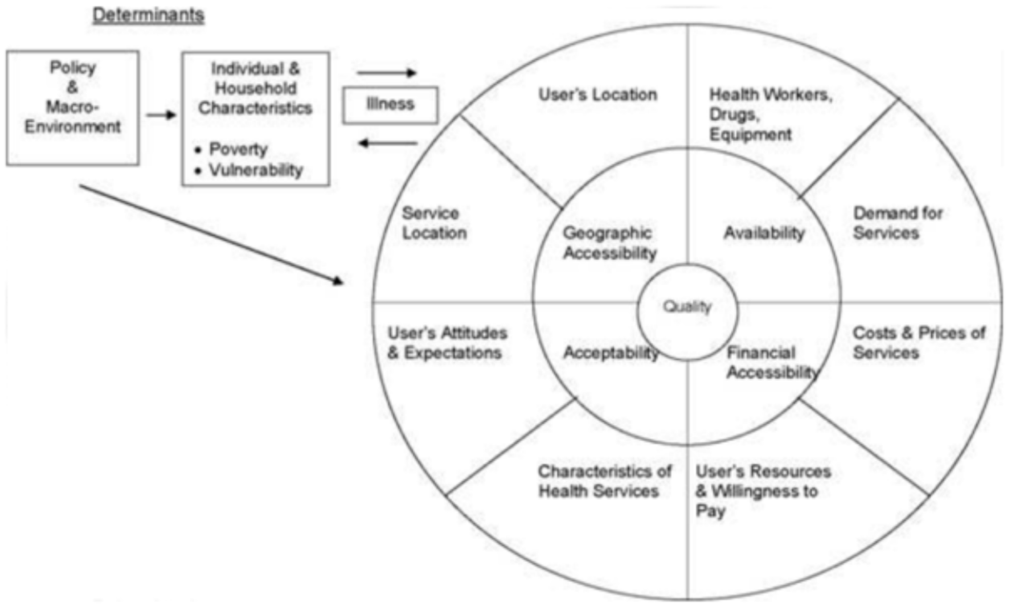

Determinants
Policy & Macro-Environment
Individual & Household Characteristics
• Poverty
• Vulnerability
Illness
User's Location
Health Workers, Drugs, Equipment
Service Location
Geographic Accessibility
Availability
Demand for Services
Quality
User's Attitudes & Expectations
Acceptability
Financial Accessibility
Costs & Prices of Services
Characteristics of Health Services
User's Resources & Willingness to Pay

Figure 1.1: Conceptual framework for assessing the access to health services.

- Geographic accessibility: The travel distance or travel time needed by the patient for getting to the healthcare service facility.
- Availability: To have the necessary care available to those who need it, such as the availability of health workers, and of required equipment at the facility.
- Financial accessibility: The costs for the services as well as the willingness and ability of the patients to pay for those services, protected from the economic consequences of health costs.
- Acceptability: The characteristics of the health services how they are perceived, and the users' attitudes and expectations from the services.

The rural and remote areas of developing countries are facing all the dimensions as mentioned above. In these areas, there is an insufficient number of doctors to meet the patients' demands. Several rural areas are without Primary Health Posts or Community

Health Posts, and even if it exists they are vacant or without any facilities. Although in some areas there is a presence of medical doctors, they are short of other staff, and all the work should be carried out by the doctors themselves, which can delay the consultation time with the patients and cause poor quality of the healthcare services. A systematic review of the average time for primary care consult in 71 countries showed that on average a primary care consult in India lasts 2.5 minutes, in Bangladesh, it is 48 seconds [19]. Hence to solve all these hurdles eHealth and mHealth will be a unique opportunity to provide primary care virtually, with the potential to improve access on all dimensions.

1.2 Mobile Health (mHealth)

The terms eHealth and mHealth are quite similar, the only difference is that eHealth implies the use of a computer as well as a network, while mHealth implies using mobile devices, portable computers, personal digital assistant devices (PDAs), and tablet PCs. Mobile health (mHealth) refers to the process of collection of medical health data of the patient in real-time and to store it for analysis, visualize of patient health trends and forward to the internet servers for further processing. Hence, the data can be used by clients such as insurance companies, hospitals, and other research companies. These data are more crucial for doctors as they can use it for the monitor, diagnose and treat patients in an effective way. Medical data are captured by the medical sensors which are attached in the body of patients in the form of a wearable watch or other wearable medical devices and saves in smartphones with the help of an application on a mobile device. Integration of these medical health devices and mobile provide the environment to predict the health anomalies of the patient in real-time which can lead to diagnosis, treatment or hospitalization in case of medical emergency [20]. Since there are many numbers of mHealth application found in application store which has a significant positive impact on health and health care, it poses a novel challenge for patients and clinician to choose proven products rather than an endless choice of unproven products. Nowadays, mHealth makes the patient be aware of themselves and help the people saving their time and money of visiting hospitals unless there is a real need for it. Mobile health will be the key to personal healthcare services in the future.

1.3 Problem Statement

Although many numbers of eHealth and mHealth application have been developed in recent years [25] [37], there are still some problems that need to be addressed. They are specially designed for the high bandwidth network, so while building a framework for these rural areas, we should take consideration of the network. They should be capable of running in low-band-with and on offline mode too for a primary onsite consultation. Most of the application is sophisticated and not user-friendly, so we should take consider to make it simple and user-friendly. The vital sign measurement process is still traditional and manually entered into the application or medical record which can cause human error. Although the medical sensors used to measure vital signs are digital and transmits data, it is difficult to receive the data in devices because profile used by healthcare devices and the profiles of the related communication interfaces are on different standards. Healthcare devices do not come in a plug and play manner and making it harder to integrate on a single application. The other main problem is that they are not user-friendly which will be a concern for the health workers with basic health knowledge and need through training to be familiar with it. Another concern is the cost which will be taken by the developers of the application to give the service and training. Also, most of them do not give early warning by taking the vital signs it merely stores the data and forward to medical personnel for review and wait for a result which can cause delay for giving basic service on-site by health workers with basic knowledge which can cause. Many of the application does not give real-time consultation and observation.

1.4 Research Questions

1. How the application support the health care service which provides decision support and recommendation with the help of technology?

2. How the application can read the data from medical sensors automatically and can reduce human input error?

3. How can we make ease of use of the application by primary health workers with basic health knowledge or with little training?

4. How to make it useful in a low-bandwidth network or in offline mode too?

1.5 Related research

Recently the rapid development of the mHealth apps in the markets has brought confusion among the users about which of the app they must rely on, this made a user a daunting task to choose the correct app. Among those ample of mHealth apps, only a small subset of apps are regulated and approved by the concerned body which also ensures the necessary standard and security of these apps. With these kinds of increase application which provide diagnosis and treatment information, medical physicians and consumers are demanding applications that can deliver safer and high-quality mHealth apps. When the smartphones came into existence a decades ago, number of available mHealth applications available in the markets are also estimated to increase substantially. A research conducted by Research2Guidance in 2018 shows that are more than 318,000 health apps which are available on the top mobile stores and daily the numbers of applications are increasing by approximately more than 200 apps [9]. These mHealth applications can help to bring a significant positive impact on health and healthcare but also pose a challenge for health workers and patients with an endless choice of mostly unproven products.

Doctors and health workers still struggle with the selection for the right choice of application which is useful and can effectively work so they can recommend to their patient without any fear. Before referring to their patient physicians should consider the authenticity and value of these applications because most of these applications are built without any involvement of medical experts and may do not follow the proper medical guidelines. The applications are being published in the market without any appropriate validation and without undergoing the trial phase [14]. A research conducted by the US and European researchers on a particular mHealth app which helps people to cope with anxiety disorders on App Store and Play Store found out 52 apps related to this. Among these applications, the majority (33/52, 63.5%) provides no information about the intervention, and more than two third of the application (35/52, 67.3%) did not provide the information about the medical standard and involvement of the consultant. Only the small number of apps about 4% (2/52) offered the information about the right standard and usability for the application [36]. In a similar way mHealth apps which relate to telehealth also have the same situation, although there are many number of applications found in the market 50% of them come with the price which asks in the form of consultation. Also, 80% of the application need the working internet connection; most of them are focused on profit-making. These applications are bound to do some limited number of the task such as symptom checkers, measure the medical parameters, healthcare professional finders, managing clinical and financial records, condition education and management, self-monitoring, remote patient monitoring, rehabilitation programs or prescription filling and compliance.

We dive into some of the applications from the play store which is related to our prototype. We came across the app called "MedM" which is remote patient monitoring application which measures the data from the medical sensors and automatically stores the data for the review, and the app is providing no onsite information about the status and no further checking for symptoms. We also found other similar application called "Intelehealth" which also provide to collect patient medical information and other data but not measuring of the data from the sensors are provided, and also no onsite information is being provided about the health conditions of patients. Also, we found many apps for personal use not dedicated to the health workers so they can record information and do a checkup of the other patient. We also found out the different mHealth application which connects the doctors for a consultation but asks for the price for the treatment. During the research of these apps from store, we came across a different application which is unable to provide necessary primary health knowledge information to the patient. We also found out many application for symptoms checking such as "Symptomate", "Ada" and many others but all require a reliable internet connection to operate. We could not find the application which is similar to our application concept.The application which can collect medical information of the patient, analysis of the vital parameters record patient family and medical history and assessment of symptoms and provide information about it and display a report for an overview and save the data for further use.

1.6 Aims and Objectives

The main aim of this project is to build an application that can be helpful for the health workers in remote areas of different countries in the world. This application will create a profile of health worker so that they can record the patient information, records there vital signs from a sensor or manually. It also helps to calculate a score which gives the information about the degree of illness of the patient and what sort of care does he need according to the score. Also, this application record family and medical history of the patient and analysis the possible finding of the medical condition by analyzing the symptoms provided by the patient by questionnaire session. Below use case diagram shows the general overview of the application which we aim to develop.

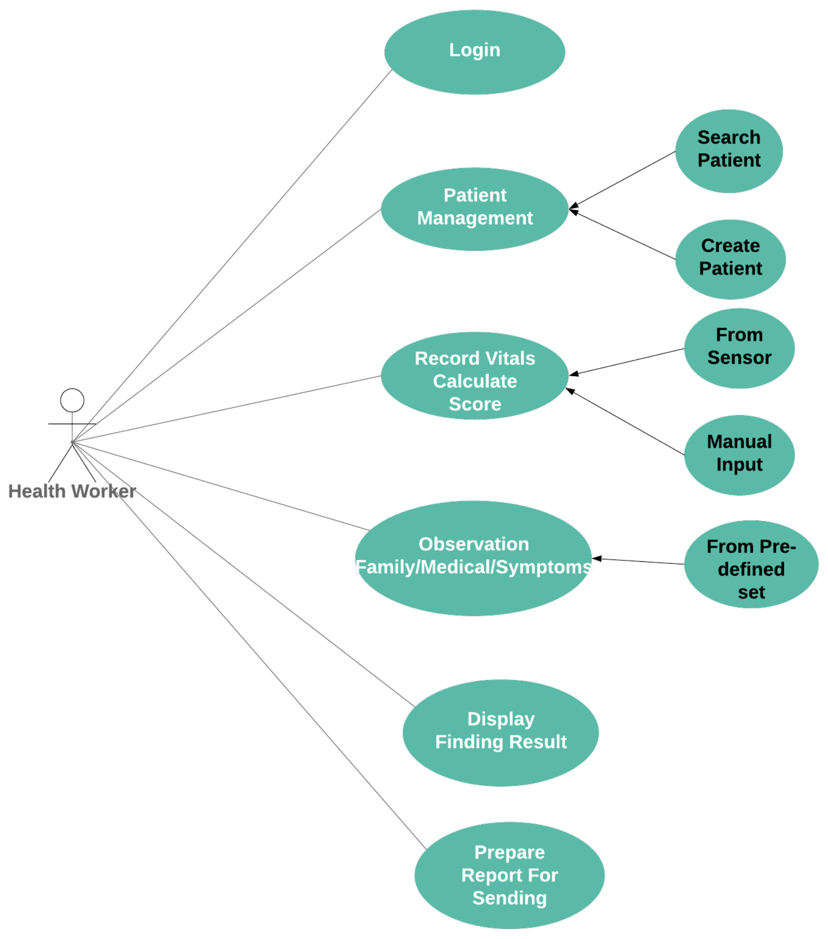


Figure 1.2: Use case diagram of application

The main objective of the development of the application is to facilitate the people in rural areas in different parts of the world with the hope they can get access some primary health service with the help of health workers. This application helps to advance the information gathering process of the patients using the mobile health platform rather than the traditional paper format. Also, to provide medical information and analysis of vital parameter and symptoms. This is achieved by creating the application which can run on mobile or tablet so that it is portable and feasible for health worker that they can carry everywhere and can start giving health service anywhere rather than sitting on the room with the large PC.

1.7 Project Approach

The main goal to achieve in this project is to develop a smart and effective application for a health worker, which can be operated by minimal health knowledge. To accomplish this task, we used the Android Studio to build the application which can run on any devices which run on Android OS. The main reason to choose Android programming is due to its popularity and low-cost devices and are readily available around the world. To save the patient information application would provide an interface where details can be manually inputted on the respective field and saved on the database. We also use the Continual certified medical devices to measure the vital parameter of the patients, and the readings can be transferred automatically by the application. To read data from medical devices, we build the Bluetooth API, which then parses the data from medical sensors to the application using the IEEE and Continua standards for data exchange [3]. The data received are stored on the local database, and with the help of vital parameter, a score is calculated which determines the degree of illness of a patient which gives an overview of how the patient should be treated further. Here with the help of application we also record the family and medical history of the patient and also with the symptom analysis tool we can get an overview of the patient health condition an how we can proceed further. All of the data of the patient are then stored in a local database and are made ready for sending to the doctor or the back-end server for further analysis or processing.

1.8 Report Structure

This thesis is divided into seven chapters as follows:

Chapter two represents the theoretical and technical background where this work is based on. It includes the review of the Continua Design Guidelines used for the medical devices to exchange the data with Personal health devices and gateway. Bluetooth and Bluetooth Low Energy technology for sending and receiving data and its working principle. We also discuss the platform used for developing an application which is Android and IDE as Android Studio. The programming language used for development are JAVA, XML, JSON, and SQLite database.

Chapter three presents the details and description of the design and solution of our application. We describe here the overview of our framework how it is designed, wireframing of the application which is the rough sketch of application which is going to be developed and also the modeling of the database how data are saved on local database tables in our devices. Further, how mind mapping is done to analysis symptoms and the way to calculate the score.Also, we discuss the about the solution of the application how the data are read from medical sensor, how score is calculated and data are saved in application and also how symptom analysis. Mainly code part of application are discussed in this chapter.

In chapter four, we discuss the results of the application how are activity screen design will be presented, how will be the flow of the application where is the input field for patient information. Also, options for symptoms analysis, information visualization such as the score of patient, family and medical history of patients and observation. In chapter five we discussed about the evaluation and testing of our application with some of the users. In chapter six and seven discussion and conclusion of the thesis is discussed.

Chapter 2

Theoretical & Technical Background

In this chapter, we present theoretical and technical background information that is relevant to the thesis. It mainly focuses on standards, application, tools and programming languages that are required for the completion of building the mTeleHealth-UiA app.

2.1 Vital Signs Parameter

Vital Sign is the measurement of the most basic function of our body. The prominent four vital sign routinely checked by medical personnel include:

1. Blood pressure

2. Pulse rate

3. Breathing rate (respiration)

4. Body temperature

The vital sign is an important parameter which helps to detect and monitors the medical problem and can be measured on any place at home, hospital, medical emergency or elsewhere [31]. They are also the essential parameters which are required for the calculation of Early Warning Score (EWS) which are the simple physiological score system. This score apprises the quick overview of the medical status of the patient to prevent the catastrophic deterioration in clinical area [32]. There are many standards for assessment of the early warning score such as National Early Warning Score (NEWS), Modified Early Warning Score (MEWS) and other standard, among them we use NEWS for assessment to obtain the score for the evaluation of the patient. It is based on the aggregation of the six physiological parameters, four of them are respiration rate, temperature, systolic blood pressure, pulse rate and the other two which are also known as fifth vital sign and are level of consciousness or new confusion and oxygen saturation [24].

2.1.1 Body Temperature

Normal body temperature of the person ranges from 97.8 F (36.5 C) to 99 F (37.2 C) for a healthy adult. The normal temperature of the body depends upon the recent activity, time of day, food fluid consumption and among other factors. There are different ways of measuring the body temperature such as orally, rectally, armpit, by ear, by the skin and internally. The normal body temperature can abnormal due to the fever which indicates a rise in body temperature by 1 degree or more or hypothermia which is a drop in body temperature[23]. The thermometer is used for measuring the body temperature and can be found on various types of classic glass thermometer, digital thermometers, special thermometer and thermometer with probes.

2.1.2 Pulse Rate

The pulse rate is the heart rate measurement which indicates the number of times of the heartbeat per minute. Pulse also indicate a heart rhythm and strength of the pulse. The normal pulse of a healthy adult ranges from 60-100 bpm, but this can fluctuate with exercise, injury, and illness and among other factor. Females do have usually higher heartbeat than the males also athletes have normally low heartbeat without any problem. Heart rate can be measured by different methods manually also by medical devices, fitness tracker and monitoring devices [22].

2.1.3 Respiratory Rate

It is a number of the breathe that we take a per minute when at rest. It is counted by measuring the number of times the chest rise per minute. It also may increase with fever, exercise, illness, and other medical conditions. Normal adult respiration ranges from 12-20 breathe per minute. It usually is measured manually, but also it is measured by medical devices [35].

2.1.4 Blood Pressure

Blood pressure measurement of the force of the blood pushed against the artery wall during the contraction and relaxation of the heart. Highest blood pressure results when heart contracts and is called systolic pressure and lowest blood pressure result when the heart relaxes and called diastolic pressure. Both pressures are measured in mm Hg(millimeters of mercury). Normal blood pressure of adult is measured as systolic of less than 120 and diastolic of less than 80 (120/80). Blood pressure can be grouped as normal, elevated, or stage 1 or stage 2 high blood pressure. It is normally measured by putting the cuff in the arm and is measured by manually using an aneroid monitor and is read by looking at a pointer or by digital monitor [13].

Apart from these, there are more vital signs which are also called the fifth vital sign, and the main parameter from these vital sign we are using for this thesis are Oxygen saturation or blood oxygen and pain. Oxygen saturation is the measure of the concentration of the oxygen circulating in the blood. It is measured in percent (%) and referred to as SpO2. Normal SpO2 values vary between 95 and 100%. Pain is also considered as a vital sign and can be categorized as two group Alert and Voice, Pain or Unresponsive. Alert means that the patient is on normal condition. Voice, Pain or Unresponsive means that the patient is in some kind of pain or he is not responding [34].

2.2 Continua Design Guidelines

The Continua Design Guidelines (CDG) are being adopted and promoted by Personal Connected Health Alliance (PCHAlliance), which is also a non-profit organization with member all around the world. PCHAlliance is the way to create a secure and interoperable health data exchange in Information and Communications Technology (ICT) framework which uses open standards for personal connected health and care. CDG offers the group of clearly outlined interfaces that provide the secure flow of medical data among sensors and gateways, removing ambiguity in underlying standards to confirm a consistent and interoperable ecosystem of personal connected health devices[2]. To ensure reliable interoperability, it implements the additional guidelines which help for further clarification of these standards and specification by reducing options in the underlying or by adding a feature missing in the underlying specification or standard. CDG uses ISO/IEEE 11073 standards for personal health device communication which make ease for a developer to build plug-and-play systems more efficiently and cost-effectively [4]. This standard family defines the optimized exchange protocol with medical devices specialization for data representation and the communication model. Typically ISO/IEEE 11073-104xx standards define the device specialization and the requirements for each medical device, such as blood pressure, pulse oximeters, thermometers and weight scales [5]. United Nations International Telecommunication Union (ITU-T) also recognize CDG and published as the Recommendation ITU-T H.810 series describing it as the safe, reliable and secure exchange of data to and from personal health devices.

Mainly CDG employs Bluetooth (Classic and Low Energy), NFC, ZigBee and USB as transport technologies on Personal Health Devices (PHD) Interface. In this project, we use PHD that are continua certified and uses Bluetooth Low Energy GATT families standards and also classic HDP Bluetooth for data format and exchange between the sensor and the gateway. Bluetooth technology is widely used for many years in health care devices as it provides a wireless link between devices and connects them. Although, the data protocol and format with this technology is proprietary there is still no best profile. So to overcome this problem, there is a necessity to adopt an interoperable wireless standard by the device manufacturer [28].

2.2.1 Architecture Overview

The main objective of CDG is to enable health devices manufacturer to certify and design sensors, gateways, health fitness server, electronic health record and to interoperate between the broad range of vendors or manufacturers. Continua mainly focus on the data that need to be transmitted between the devices on the application protocol and transport layer and maintain interoperability between them. So to maintain this interoperability, it defined a standard architecture based on the four component as shown in figure 2.1

- The Personal Health Devices (PHD)
- The Personal Health Gateway (PHG)
- The Health & Fitness Server (HFS)
- The Health Information System (HIS)

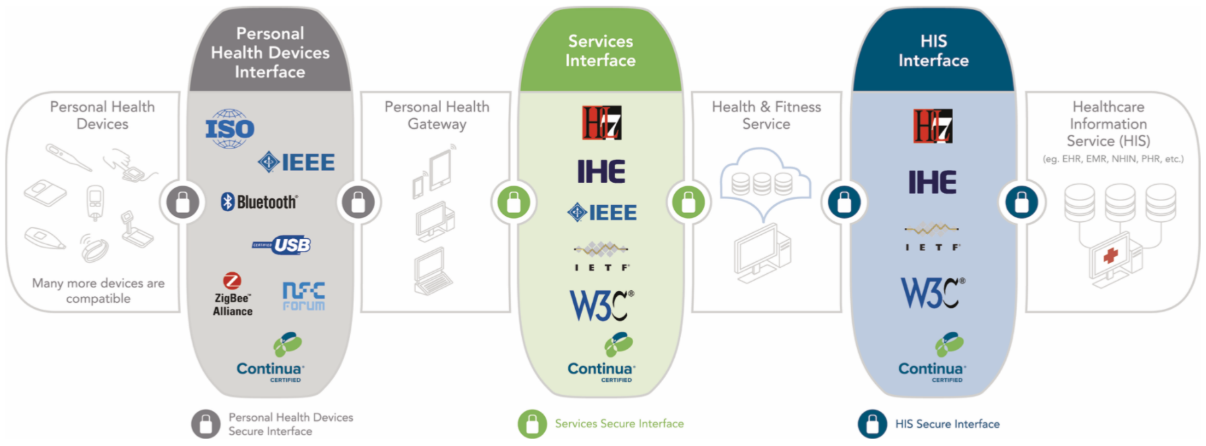


Figure 2.1: Continua E2E Reference Architecture

,

Here in our application we use the first two component PHD and PHG the other components can be used in the future for the extension of this project. As we know from figure

PHD are devices such as heart rate monitor device, blood pressure device, a temperature device, weight scaling devices, personal alarms, and PHG are devices such as mobile phones, tablets, PC or specialist device. Also, the other two components HFS and HIS are typically a telehealth remote monitoring service platform and a physician's electronic health record respectively. Here CDG set the requirement for the interface to communicate between these components. For the communication between PHD and PHG, PHD Interface employs the IEEE 11073 PHD family of standards for data format and exchange between them using USB, Bluetooth, BR/EDR, NFC BR/EDR and ZigBee. Similarly, to communicate between PHG and HFS, it uses the service interface that employs SOAP for message packing and authenticates using HL7 standards also HIS interface uses HL7 standards such as CDA, R2, HL7 FIHR based Personal Health Monitoring Report for to communicate between HIS and HFS (e.g. EHR, EMR) [3]. By doing these continua helps to maintain end-to-end security and privacy through identity management and authentication of the components and interfaces.

2.2.2 Personal Health Device

Nowadays personal health devices are being ubiquitous and can be found on different types and models which makes ease to measure various health parameters also to monitor and visualize in a comprehensive way. These devices are equipped with their respective sensors, which are used for to measure required data and are being transported or exchanged between the PHG with any one of these transport technology Near-Field Communication (NFC), USB, Bluetooth Basic rate (BR)/Enhanced Data rate (EDR), Bluetooth Low Energy (LE) or ZigBee.
For example, pressure sensors may be used in blood pressure monitors to measure blood pressure. These sensors can support either of this transport technology or all of them, and the above table shows the health sensors till date following these guidelines with transport capability [3].

Although we can find a variety of PHD in the market which can be used to measure a

Capability	Transport				
	USB	Bluetooth BR/EDR	Bluetooth LE	NFC .	ZigBee .
pulse oximeter	Yes	Yes	Yes	Yes	Yes
blood pressure monitor	Yes	Yes	Yes	Yes	Yes
thermometer	Yes	Yes	Yes	Yes	Yes
weighing-scales	Yes	Yes	Yes	Yes	Yes
glucose meter	Yes	Yes	Yes	Yes	Yes
step counter	Yes	Yes		Yes	Yes
cardiovascular fitness	Yes	Yes		Yes	Yes
strength fitness	Yes	Yes		Yes	Yes
motion sensor	Yes	Yes		Yes	Yes
dosage sensor	Yes	Yes		Yes	Yes
temperature sensor	Yes	Yes		Yes	Yes
activity hub	Yes	Yes		Yes	Yes
PERS sensor	Yes	Yes		Yes	Yes
CO sensor	Yes	Yes		Yes	Yes
gas sensor	Yes	Yes		Yes	Yes
heart rate sensor	Yes	Yes	Yes	Yes	Yes
basic 1-3 lead ECG sensor	Yes	Yes		Yes	Yes
body composition analyser	Yes	Yes		Yes	Yes
SABTE	Yes	Yes		Yes	Yes
INR meter	Yes	Yes		Yes	Yes
continuous glucose monitor	Yes	Yes	Yes	Yes	Yes
insulin pump monitor	Yes	Yes	Yes	Yes	Yes
adherence monitor	Yes	Yes		Yes	Yes
peak flow meter	Yes	Yes		Yes	Yes
fall sensor	Yes	Yes		Yes	Yes
enuresis sensor	Yes	Yes		Yes	Yes

Table 2.1: Different PHD with their transport capability following CDG

different type of health parameters for medical diagnosis. Here in our thesis we use three health devices which are available to us, Blood Pressure Monitor, Health Thermometer and Pulse Oximeter and are capable of measuring almost all type of vital signs. Vital signs represent the most important body signs which indicate our body vital function. Normally there are four important vital signs blood pressure, body temperature, heart rate, and respiratory rate. There are a lot of other signs such as oxygen saturation, pain, blood glucose and many more. These signs are measured by health personnel to get an early overview of the patient and how to treat them accordingly. We will be discussing vital sign and Early Warning Score later on this report. The three PHD which are used for to measure vital signs are A&D Medical UA-651BLE Upper Arm Blood Pressure Monitor with Bluetooth Smart/ Bluetooth Low Energy Connectivity which is capable of measuring blood pressure and heart rate which is shown in figure 2.2.

Figure 2.2: A&D Upper Arm Blood Pressure Monitor

Similarly, for the measurement of the body temperature we use A&D Medical UT-201 with Bluetooth Smart/ Bluetooth Low Energy Connectivity as shown in figure 2.3

Figure 2.3: A&D Digital Thermometer

Also, the third one which is used for measurement of blood oxygen and heart rate is Nonin Onyx II 9560 Fingertip Pulse Oximeter W/ Bluetooth as shown in figure 2.4.
Among these devices, blood pressure monitor and digital thermometer use Bluetooth Low Energy (Bluetooth LE), and pulse oximeter uses Bluetooth Classic as a transport technology for to send data from the device to the gateway also use design guidelines referred by Continua. We will be discussing in the implementation section below in this report how we programme to receive data in gateway from the device.

Figure 2.4: Nonin Onyx II 9560 Pulsoximeter

2.2.3 Personal Health Gateway

The reliable connectivity of personal health devices mainly depends on a health gateway that links the patient and remote caregiver. This health gateway can be found in the different form of devices such as smartphones, tablets, laptops, notebooks, personal computers and also specially designed devices to perform a designated task. Also, these type of devices runs on different OS had different hardware specification which we can choose accordingly to our needs and requirements. Nowadays smartphones and tablets are being emerged as a leading platform and first choice for implementing personal health gateway[29]. Due to portable computation and innovative mobile communication in smartphones and tablets, it is capable of running third-party software which causes a rapid increase in their use among healthcare professional. Many medical application is being developed these days targeting health professionals and patient. The usage of smartphones and tablets are getting more attention in the field of health care day by day. Smartphones can play a significant role in patient health care observation, measurement of different vital signs, disease self-management and monitoring of patient [27]. Therefore smartphones and tablet are being popular health gateway nowadays and have been integrated into healthcare practice areas due to ample quality medical apps.

Although smartphones and tablet are popular these days, we should also take consideration of the OS or platform on which these runs. There are a lot of OS such as Android, iOS, Windows, Blackberry, SymbianOS and many others. We should also consider the cost, availability, and ease of use of these smartphones and tablets which operates on these OS while choosing them to make a gateway. Among these Android OS is the most popular, low cost and readily available to build third-party apps. According to the data provides by StatCounter GlobalStats it shows that Android is the most popular OS and got the highest market share worldwide. Below figure 2.5 shows the market share of different OS in the period of last one-year [8].

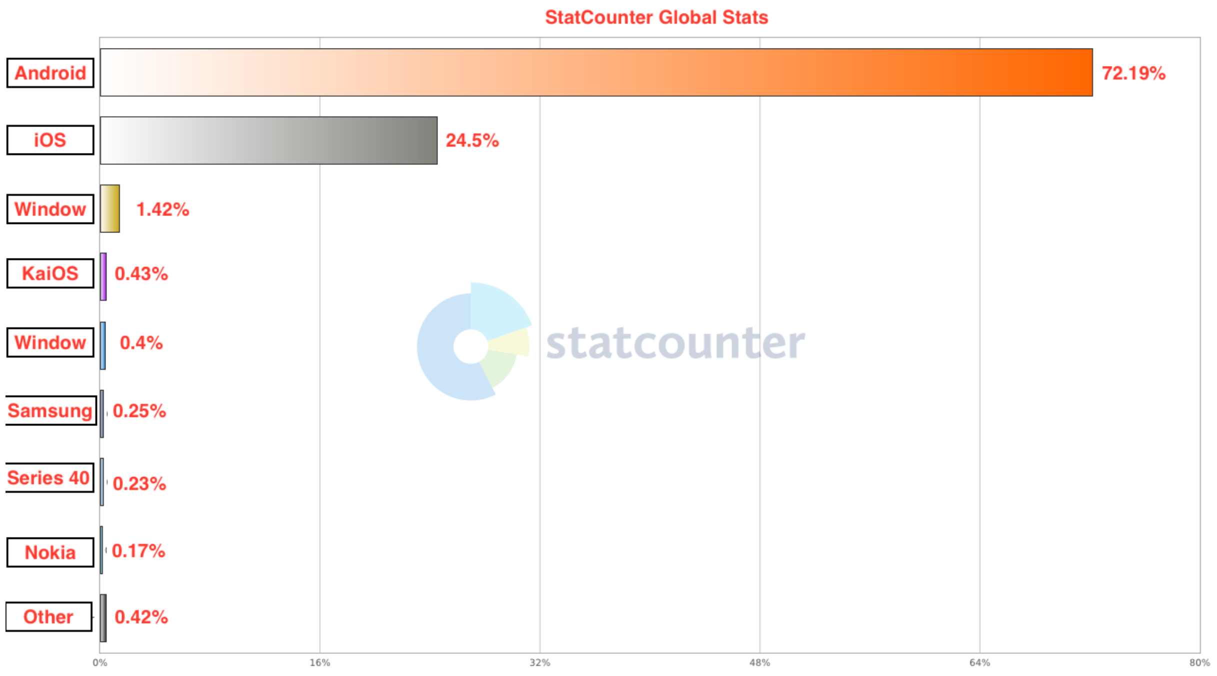


Figure 2.5: Mobile & Tablet Operating System Market Share Worldwide

,

Hence due to its popularity, low cost and ease of availability in every part of the country either in urban or rural areas we are going to make a mHealth application which runs on Android OS platform and can be installed on smartphone or tablet which acts as a gateway to collect and store the data. Although we can choose from a wide range of smartphones and tablet which is compatible with this application, we select Samsung Galaxy Tab S2 9.7 VE SM-T819 model in our development process which is shown in the figure 2.6 below.

Figure 2.6: Samsung Galaxy Tab S2

,

2.2.4 Personal Health Device Interface

PHD interface defines a framework for personal connected health services to ensure interoperability between the devices and the exchange of the data smoothly between them. It usually implements a specific module such as NFC Interface, USB Interface, ZigBee Interface, Bluetooth Interface, Bluetooth Low Energy Interface. The collaboration between PHCAlliance and IEEE sets out a standard of IEEE 11073 Personal Health Device family to address interoperability of these personal health devices (e.g. health thermometer, blood pressure monitor). The family standards also ensure the user that what is measured from where and how it is measured and is being transported correctly without loss of data to electronic health record via health gateway [2]. Here in our project, we use health devices which uses Bluetooth Interface for Nonnin Oximeter and Bluetooth Low Energy Interface for blood pressure monitor and health thermometer. We can also find other interfaces for the same type of devices and also many other devices with a variety of interfaces too which are shown on the above table 2.1 with their transport capabilities.

Bluetooth and Bluetooth Low Energy Interface (BLE)

The connectivity of Bluetooth interface ensures the basic requirements which are set for all CDG-certified products:

- bidirectional sensor control

- bidirectional sensor information exchange
- appropriate linkage between a Personal Health Device and an Personal Health Gateway

We can see from the figure 2.7 that this interface is structured into three typical layers with appropriate standards for selected individual layer in the personal health ecosystem to maintain interoperability. It also defines the guidelines for scanning, discovery, pairing, notifications, secure pairing, and quality of services [6]. These services will be discussed below later on the implementation section.

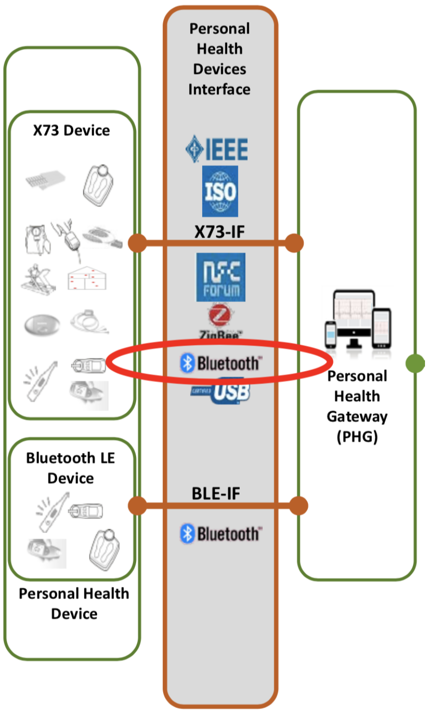


Figure 2.7: Bluetooth interface

BLE is also Continua supported transport technology and a part of PHD Interface which is widely accepted nowadays in PHD due to its low power consumption. However, on the other side it has a low-bandwidth and limited range of wireless connectivity. The specific profiles are defined by Bluetooth Special Interest Group (BT-SIG) and operate on top of the Bluetooth Low Energy Attribute Profile which is supported by PHD interface. Similar as Bluetooth Continua also defines the guidelines for BLE for the services discovery, scanning, pairing, secure connection, notification, device information, date and time information, certification and regulatory aspect and transcoding for interoperability and security.

2.3 Bluetooth

Bluetooth is a global wireless communication standard for wirelessly connecting devices such as mobile phones, computers, and peripherals over a certain distance to transmit voice or data, and it operates in the same 2.4 gigahertz frequency as Wi-Fi [11]. Bluetooth is just a way to eliminate all wires and cables that normally connect devices while still keeping the communication between them secure. Bluetooth was developed back in 1994 by Jeff Hudson 1990s while he was working at Ericsson. However the origins of the name Bluetooth date way back to the 10th century in Denmark it was named after Danish king Harald Bluetooth. It is commonly referred to as piconet and uses master/slave model, master from the word itself refers that it coordinates all communication throughout the piconet it can send or request data from its slave device. In this model, a single master device can have up to seven slave devices connected to it while a slave device can only be connected to a single master device. Slaves are allowed only to send and receive data from the master and are not allowed to talk to each other as shown in figure 2.8 below.

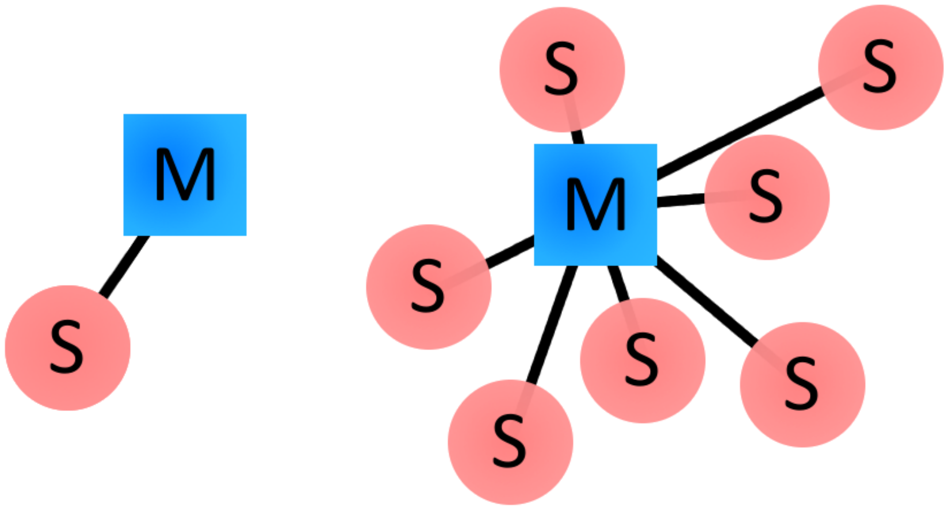


Figure 2.8: Examples of Bluetooth master/slave piconet topologies.

Bluetooth is divided into three class by its range each class has its max transmission range, class one has a max range of 100 meters, class two has ten meters, and class three can transmit up to one meter however we have been able to surpass the hundred-meter mark with the introduction of Bluetooth 5.0.

Bluetooth has a series of generations the first generation of Bluetooth is pretty much non-existent right now unless we have a really old device its speeds are about 700 kilobits per second. The second generation introduced faster data rates at around 2.1 megabits per second with enhanced data rate and NFC communication, and it also cut power consumption in half. Third generation Bluetooth 3.0 or high-speed Bluetooth allows for a transfer rate of up to 24 megabits per second. This is made possible by using an 802.11 link for a large data transfers, so the Bluetooth link itself is not used for large data transfers instead it is used for negotiation and coordination while the 802.11 link handles the large data transfer. The fourth generation Bluetooth offers lower power consumption with its low energy technology which is also known as Bluetooth Smart. It enables devices to remain paired without requiring a continual stream of data between devices. Low energy technology also allows Bluetooth to be used in low powered devices such as heart rate sensors thermometers Fitness sensors and many more as for speeds they are roughly the same with that of Bluetooth 3.0 at about 24 megabits per second. Bluetooth 5.0 will have four times the range twice the speed and eight times the data broadcasting capacity of the previous generation the 5.0 upgrade is mainly catered towards the emerging Internet of Things market faster speeds, and larger transmission ranges mean that Bluetooth can now be an option to replace the more power-hungry Wi-Fi in IOT applications.

Bluetooth standard is maintained by a group of commercial electronics companies known as Bluetooth Special Interest Group (SIG). These groups define the Bluetooth profile and possible applications of each profile. In order to use Bluetooth in the device, it must be compatible with the desired Bluetooth profiles to use its services. The profiles are the specification which defines Bluetooth-based wireless communication between devices and resides on top of the Bluetooth core specification. The profile capabilities define how the device can use the Bluetooth technology and provides standards to the manufacturer to use it in intended manner. The Bluetooth SIG group defined many profiles according to the possible applications of each profile. Some of the profiles are Advanced Audio Distribution Profile (A2DP) which defines how multimedia can be stream from one to another device over Bluetooth. Health Device Profile (HDP) defines the transmission and reception of health data from devices over Bluetooth, Generic Attribute Profile (GATT) specially provides services for Bluetooth Low Energy protocol it is set of Attribute Profile (ATT) to form services.

2.3.1 Classic Bluetooth for Android

Classic Bluetooth network stack support has been introduced in Android platform which allows the devices to exchange the data wirelessly with other Bluetooth devices. Android Bluetooth APIs lets the application framework for providing access to the Bluetooth functionality. These Bluetooth APIs helps Android application to scan other Bluetooth devices, to query local Bluetooth adapter, the establishment of RFCOMM channels, connection of devices through service discovery, transferring data from and to other Bluetooth devices and managing the multiple connections. Classic Bluetooth is the right choice for the operation which consumes the more battery power which includes the continuous streaming of the data and communication between the Android devices. In order to transmit data between two Bluetooth-enabled devices first, they should establish a channel for communication through *pairing* process. One device is in the state of *discoverable device* which accepts incoming connection, and another one is in *service discovery process* when service discovery finds the discoverable device it sends connection request which is accepted and paired with security keys. Here *discoverable device* are PHD with Bluetooth and *service discovery process* are run by Android device with Bluetooth. After pairing, they are ready for the exchange of information between devices after the information exchanging completes there is a release of channel and device got disconnected. These devices can be automatically bonded in future if they are within the range because of the pairing in the previous session where the device has cached the security keys, and automatically connection takes place until these keys are not removed.

Bluetooth classic also defines many profiles, but our concern is for the Health Device Profile. Android 4.0 (API level 14) start to implements support for HDP profile which helps to create an application using Bluetooth that interacts with the health devices which support Bluetooth for exchange of data and information between them such as Spo2 monitor, heart rate monitor. In using the Bluetooth API following term of HDP is the key concept:

- Source : Source are the PHD such as thermometers, heart rate monitors or gulcose monitor that are used to measure data and transmit to the smart devices such as Android tablet or phone.
- Sink : These are the Android devices which are used to receive medical data from the source. In Android HDP it is represented by *BluetoothHealthAppConfiguration* object.
- Registration : It is the process to register sink to the particular source for communication.
- Connection : It is the process to create the channel between PHD (source) and Android device (sink)

The figure 2.9 shows the overview of the HDP Manager Module which contains PHD as a source and smart health gateway as a sink and registration and connection too place between these devices and exchange of data took place using HDP profile. The process of exchange of data is discussed in the implementation section.

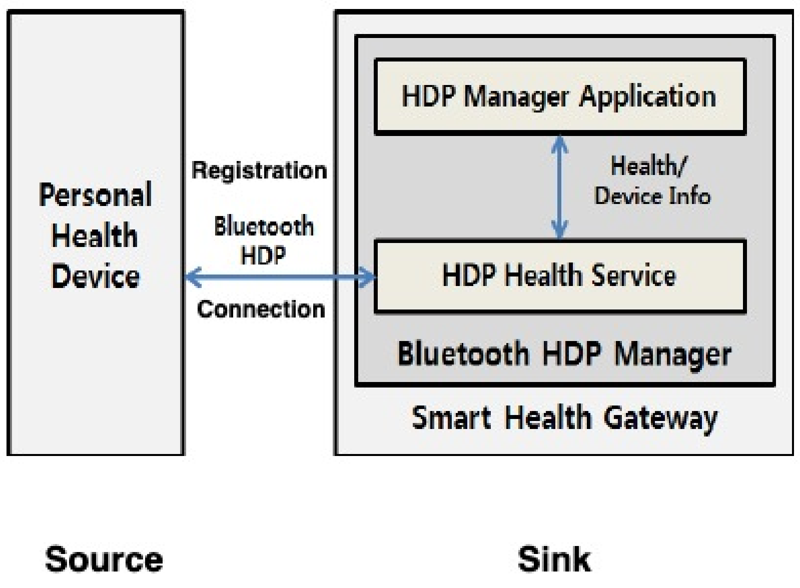


Figure 2.9: HDP Manager

2.3.2 Bluetooth Low Energy for Android

Bluetooth LE built-in platform support has been introduced in Android 4.3. Low energy consumption is one of its main characteristics compared with Bluetooth classic. Therefore, Bluetooth LE devices last for weeks, also some of them for years before needing them to recharge or replace the battery. The protocol itself is optimized for a small burst of data exchange which is ideal for applications such as sensors, remote control. In contrast Bluetooth classic quite often is involved in high bandwidth applications such as audio streaming. Below figure 2.10 shows how information is structured in BLE applications.

BLE is based on GATT profile specification which specifies how to send and receive small pieces of data which is known as attribute. The android app can either be a GATT server or GATT client according to the need of the application. The GATT server support group of services which are also the features offered by the devices.

- **Service** : Each of the services is identified by the 128-bit number known as Universally Unique Identifier (UUID) and consists a collection of one or many characteristics which can perform read or write operations according to the profile we are working. For example, we can have a "Health Thermometer" service that have char-

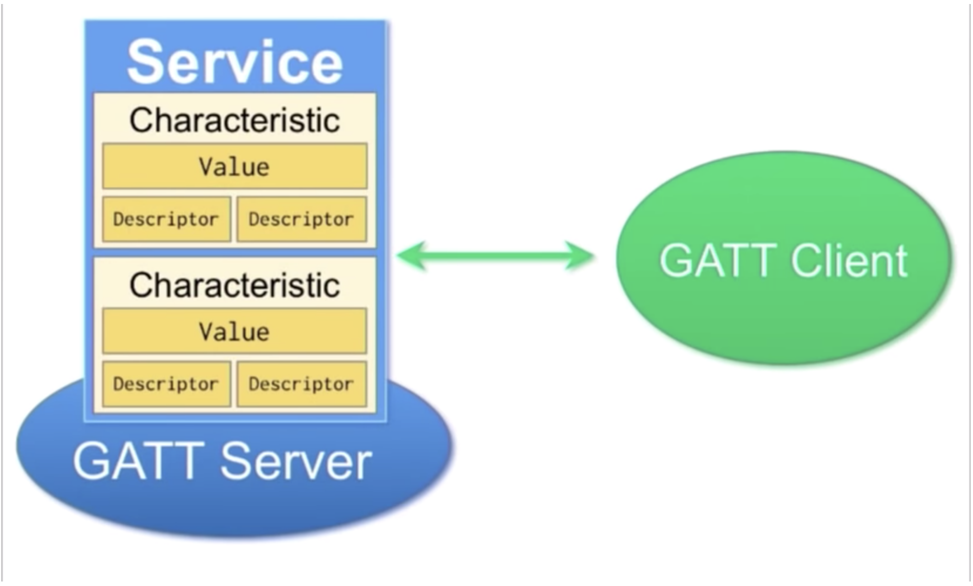


Figure 2.10: General Attribute Profile

acteristic Temperature Measurement, Temperature Type any many others which can be found on the official Bluetooth website.

- **Characteristics** : It contains a value field and one or many descriptors which are used to describe the characteristic value. They can also be seen as a type similar to a class.
- **Descriptor** : The descriptor is being used to describe characteristic value. It may be a description which is readable by humans, the characteristic value within an acceptable range, or a unit that is specified to the original characteristic value.

When an Android device and the BLE device interact with each other, there form some roles and responsibility that applies.

- **Central vs. peripheral** : BLE connection itself applies for this role, here the central device scans for the new devices and listens for the advertisement by new devices. The role of the peripheral is to make an advertisement.
- **GATT server vs. GATT client** : The role of the server and the client is to govern how devices talk to each other after the connection is initiated.

For to understand the above concept let us choose an Android tablet which supports BLE and acts as a central role and a Health Thermometer device as the peripheral role. To initiate the BLE connection we need both of them if both devices being the central or peripheral role it cannot initiate the connection, so we need each one of them either to be central or peripheral role. Once the tablet and the Health Thermometer is connected, they started exchanging GATT metadata to one another. Depending on the data they are exchanging the device act as a server or client. For example, if Health Thermometer wants to send data to tablet it act as a server but if it wants to receive some updates from the tablet, then the tablet acts as a server. To be clear, we can see the figure 2.11 below to know the BLE role in pre-connection and post- connection.

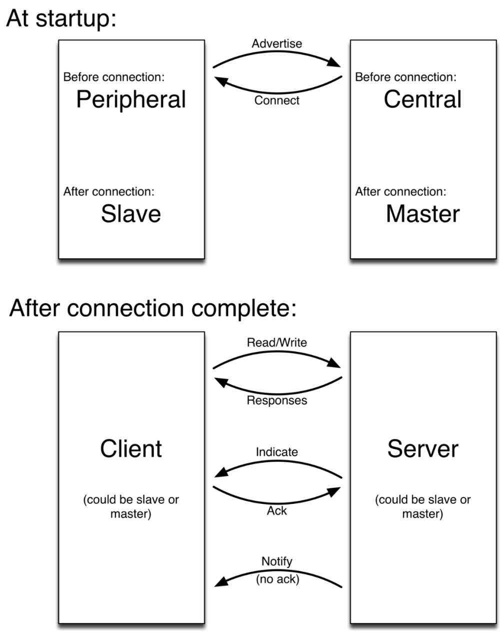


Figure 2.11: BLE roles

As we can see, the role of BLE can be divided into two parts at the startup and after the connection is established.

Pre-Connection : This is the initial phase where the device can be either central or peripheral.

1. Peripheral are usually a PHD device such as Blood pressure monitor and Health Thermometer and advertise itself in an interval of time and waits for the connection from the central.

2. Central are usually PC, mobiles and tablets which scans for other devices and con-

nect with them if available.

After the connection is made the peripheral is called slave and the central is called master.

Post-connection : This is the phase after the connection has been established, and a device can be either a server or a client.

1. The client is usually the master but it can be either master or slave, and it access the remote resource.

2. The server is usually the slave, but it also can be either master or slave. It is a database of resource which contains profiles, services, and characteristics and provides the resources to the client.

3. Here the client sends the operations such as read and write to server and server responds with data and if appropriate it also changes local data.

4. The server can also send data to the client by using indicate and notify operations without using read and write operation. Notify operation are not acknowledge by client whereas indicate are acknowledge

The further process how to read actual data from the PHD are been discussed in the implementation section.

2.4 Android

Android is an operating system mostly used for mobile devices and tablets which is initially developed by Android Inc. and was later acquired by Google in 2005 [18]. It is based on the modified version of the Linux 2.6 kernel with the collaboration of Google and members of Open Handset Alliance (OHA) for design, development, and distribution of Android. Android Open Source Project (AOSP) is now the body to govern development cycle and maintenance of Android, and most of its code is released under Apache License, which means anyone can use it by downloading its full source code. Although Android is modified version of Linux 2.6 kernel environment there, need to be many changes on several drivers and libraries, either they have been modified or created newly so that it can run on Android phones or tablets efficiently and as effectively. The Android community had developed their c library (Bionic), and develop Android java run time engine known as (Dalvik Virtual Machine DVM) due to some licensing issues. Due to the limited availability of the resource on mobile devices focus has been shifted towards optimizing the infrastructure. The Android-specific framework has been designed and implemented and it can be viewed as a complete solution stack, which incorporates the OS, middleware components and applications and modified version of Linux 2.6 kernel acts as the Hardware Abstraction Layer (HAL) [17]. The overall Android operating environment can be summarized as below:

- It is the open platform for mobile development.
- It is design based on hardware reference for mobile devices.
- It is modified Linux 2.6 kernel powered system.
- It is a runtime environment.
- It is a framework based on application and user interface (UI).

The Android system architecture is shown below in figure 2.12 and is divided into five layers application, application framework, libraries, Android run-time, and Linux kernel. The modified Linux kernel is low-level tools and operates as HAL, which provides a different device driver and manages memory and process along with networking functionality. The libraries have been interfaced from java language, and also Android specific libc (Bionic) resides here. Android runtime layer consists of the core libraries which includes core java library and IO, also holds Dalvik Virtual Machine (DVM) and provides most of the functionalities which are available in android which are being provided by the core libraries. Application framework layer relates to the API interface and the activity manager here looks after the application lifecycle. The content provider functions as the medium to get the data from other application or to share the data to other application. The notification manager on the application enables to notify custom alert and resource

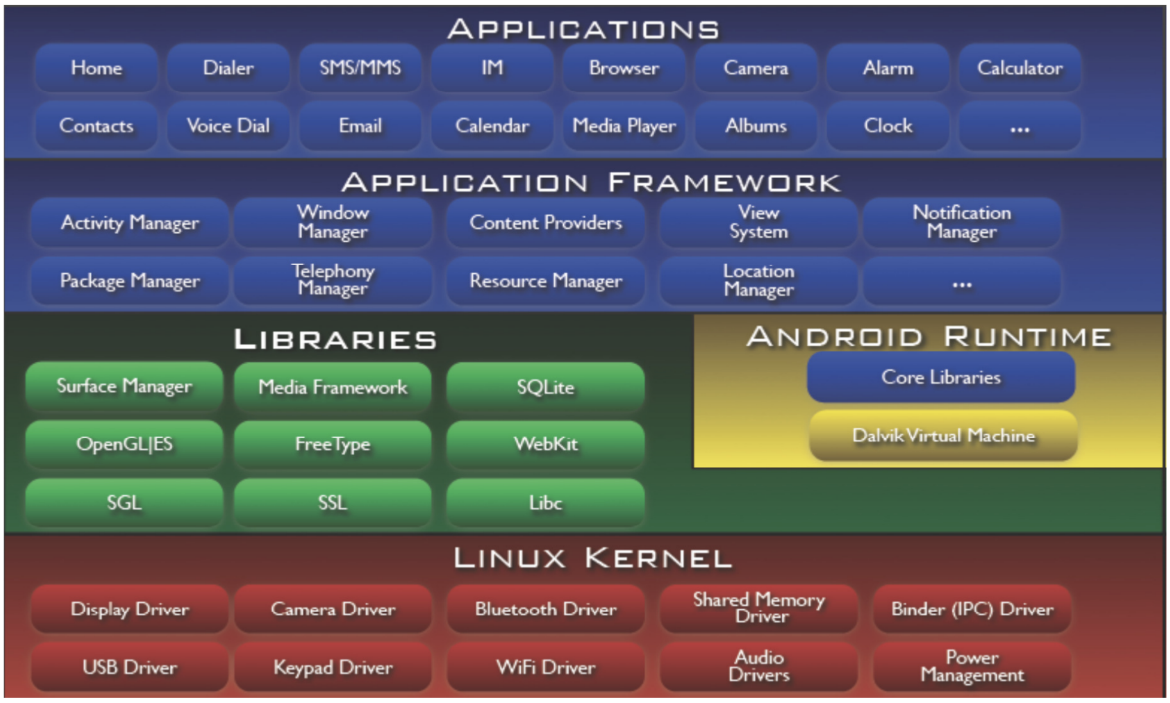


Figure 2.12: Android Architecture

manager mainly handle the graphics. On the top of the layer there resides the application layer that consists Android application which has its own process space within DVM instance [18].

2.4.1 Dalvik Virtual Machine

Android instead of using native VM of java it utilizes its own VM which is known as Dalvik Virtual Machine (DVM). The native JVM produce a java byte-code which we cannot run directly on Andriod devices. Hence we need to produce special byte-code for to run on these devices where DVM comes on action to create special byte-code. In DVM the java source code are compiled into Java bytecode, and then java class file is converted into dex file are an executable format which is optimize for mobile platform with slow CPU

limited memory resource and battery capacity and no swap space for OS. Above figure 2.13 shows how the java code is converted to be able to run on DVM. It has also been implemented in such a way that allows execution of multiple VM by the device also able for potential threading and memory management functionalities for low-level [16].

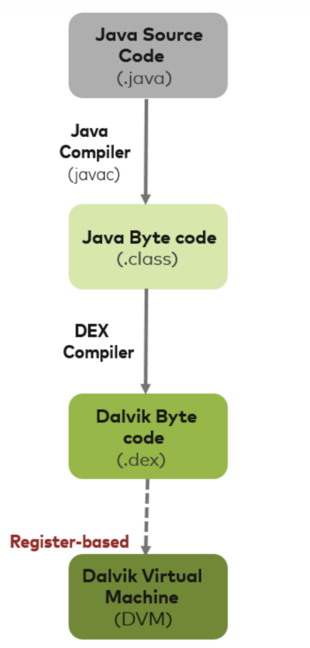


Figure 2.13: Dalvik Virtual Machine (DVM)

2.4.2 Android Application

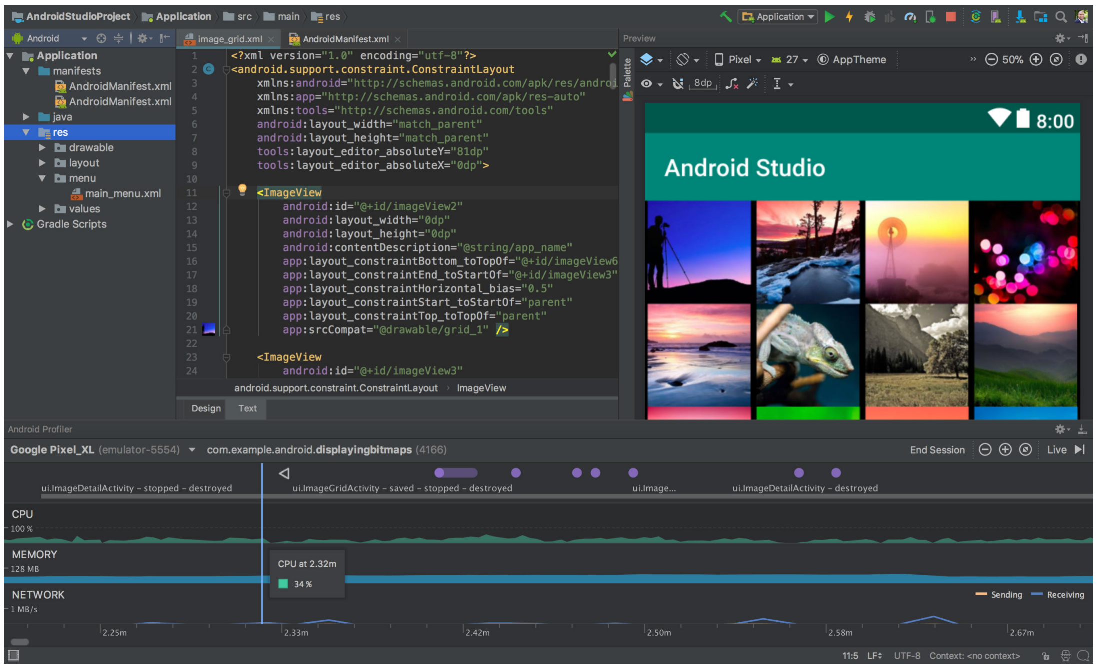
Android application files are being bundled on one single application package with a .apk extension known as Android package. We can find different android developments tools on the market, but in this thesis, we are using Andriod Studio which provides the conversion of the class file to dex file and finally to apk file which is ready for installation on devices. Android application is basically composed of activities, intents, service and content provider.

- Intents helps us to switch between activity.
- Services are the main backend of the application and execute on the background.
- Content provider deals with the sharing of the data with applications
- Activities are the screens which help us to interact with the application with a different form of UI.

To build an Android application, we need the development tool, and here we are using Android Studio. It supports two programming language Java and Kotlin for the development, but we prefer Java and develop our application using Java code. XML is also used here to design screen layout of each activity in the application and are described in the below section.

2.5 Android Studio

Android Studio is an official IDE (Integrated Development Environment) for developing application built on JetBrains IntelliJ IDEA software and designed purely for developing apps for Android devices. It is available for all the major operating systems making its availability worldwide and readily accessible. Google launched Android Studio on May 16, 2013, on Google I/O Conference event for 2013. Since then, Android Studio has replaced Eclipse and become the official IDE for developing Android applications [15]. Below figure 2.14 shows the interface of the Android Studio version 3.3 which we used to build our application.


```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/androi
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:layout_editor_absoluteY="81dp"
    tools:layout_editor_absoluteX="0dp">

    <ImageView
        android:id="@+id/imageView2"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:contentDescription="@string/app_name"
        app:layout_constraintBottom_toTopOf="@+id/imageView6
        app:layout_constraintEnd_toStartOf="@+id/imageView3"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:srcCompat="@drawable/grid_1" />

    <ImageView
        android:id="@+id/imageView3"
```

Figure 2.14: Android Studio Screen

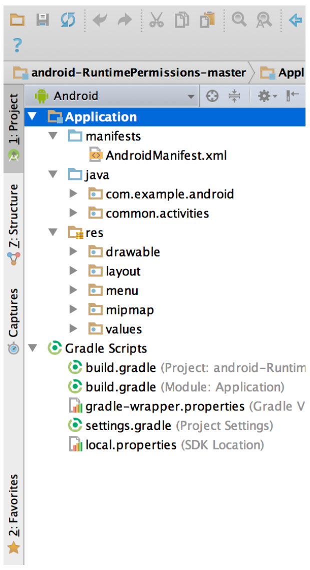
Among the lot of features of Andriod studio following features are the most current stable features :

- Build support are based on Gradle.
- Refactoring and quick fixes specific to Android and Git integration.

- Lint tools for to lookout the performance, version compatibility, usability and other problem.
- Templates for to create common design and component.
- A rich layout intelligent code editor for to write code, drag n drop UI, preview layout option and also multiple screen configuration.
- It also support for to build Android wear apps.
- It also supports for Firebase cloud messaging integration also Google map engine.
- It contains Virtual Device as a emulator to test and debug instantly faster than real physical device.

2.5.1 Project Structure in Android studio

Every project created in Android studio contains one or many modules with resource file and source codes files. The types of modules are Library modules, Android app modules, Google App Engine modules. Below figure 2.15is the Android default project view containing several project files. These structure views are for to provide quick access to any files in our project.



```
Application
    manifests
        AndroidManifest.xml
    java
        com.example.android
        common.activities
    res
        drawable
        layout
        menu
        mipmap
        values
Gradle Scripts
    build.gradle (Project: android-Runtim
    build.gradle (Module: Application)
    gradle-wrapper.properties (Gradle V
    settings.gradle (Project Settings)
    local.properties (SDK Location)
```

Figure 2.15: The project files in Android view.

All of the app modules should contain at least build files under Gradle Scripts, the manifest folder which contain AndroidManifest.xml file, java folder composed of java source code files and JUnit test files and lastly res folder which contains images file, layout files, UI string value file, and many others. Android app module structure varies from the actual project file structure which is on the disk. Also, the library module contains reusable code and had an Android Library and Java Library, and Google App Engine modules contain Google Cloud backend code [1].

2.6 Java

Java is a high-level computer programming language used to develop a platform-independent application and is class-based and object-oriented programming. James Gosling is the founder of java under the company Sun Microsystem in 1996 and is free and open source project. Java works on the principle write once run anywhere which means we write them in Java code once and can run our application on any platform. So this is called the platform independent. Java is preferred as a programming language in Android Studio IDE over others, such as C, C++, Scala, Kotlin due its fame and popularity. It has got a broad base of library and development tools, and developers are already used to it do not have to learn a new language again. It runs on VM so no need for a compilation of code every time it runs, and it is safe and secure. Although it uses java code, JVM is not used to run the applications Android uses its DVM for this purpose optimized to support for mobile devices.

2.7 Extensible Markup Language (XML)

XML is an Extensible Markup Language which is much alike as HTML, generally used to describe the data. Like HTML the XML tags are not predefined and has to be pre-defined by Spourself. XML codes are clean and simple and readable by both human and machine also simple to develop and scalable. We use XML in Android Studio to design our layout because of its lightweight capability and not making our layout heavy. Each

Android application UI or activity screen is made up of a combination of Android View and Android ViewGroup. View is a base class of the UI component. Some of the common view subclasses are TextView, EditText, Button, CheckBox, RadioButton, ImageButton, Progress Bar and Spinner. ViewGroup is also a subclass of View and a base class for layouts and contains invisible container for holding the Views. Most common used ViewGroup is Linear Layout, Relative Layout, Table Layout, Frame Layout, Web View, List View, Grid View. For example, LinearLayout is ViewGroup which hold UI control like Button, Text, Spinner and many more.

In Android, there are several categories of XML files used, and each has a different purpose:

- Layout XML Files: It is the file that defines the code which is actual UI of the app, and these files reside inside app/res/layout folder in Android app structure view.
- Manifest XML File(Manifest.xml): It is the file which defined all the component of our application needs, such as application package, Activities, services, receivers, and the permissions. For example, if we need internet service for us, we need to define it in this file. These files reside inside app/manifest folder in Android app structure view.
- Strings XML File(strings.xml): It is the file which replaces all the hard-coded strings in our activity or layout file with one string file. It helps use the reusability of code and reside inside app/res/values/ folder in Android app structure view.
- Styles XML File(styles.xml): It is the file which defines the different looks and style for our application UI. We can define our custom style and reside inside app/res/values/ folder in Android app structure view.
- Drawable XML Files: It is a file which defines various graphics for the elements or view of the application. For example, if we need different color in the background of buttons, we define inside these files. It resides inside an app/drawable/ folder in Android app structure view.

- Color XML File (colors.xml): It is the file which is used to define all the colors used in our app, define here and use in-app. It resides inside app/res/ folder in Android app structure view.

- Dimension XML File(dimens.xml): It is the file which is used to define the dimensions of View. For example, if we need a button with 50-pixel height, we define here value 50 and use in our app. It resides inside app/res/ values folder in Android app structure view.

2.8 SQLite in Android

SQLite is an opensource structure query base database and is lightweight with no need for network access standalone database. The data are saved on the text files on the device. SQLite database is already implemented on every Android device and support embedded relational database features. It holds lightweight data and is already compatible with may programming language, so there is no need for initial setup or administration procedure required. SQLite is preferred when a device needs to save a large amount of data than other repository systems like SharedPreferences. In device created database is saved under the data/data/APP_Name/databases/DATABASE_NAME.

In SQLite, the major operation is onCreate which are used for creating the database for the first time and only run once throughout the application lifecycle, and onUpgrade is called whenever there is some update on the existing database version. There is also other four query function which is Insert, Read, Delete & Update Operation. Insert operation perform to add value to the database. Read is used for reading already saved data from the database, Update for updating and delete for deletion of data from the database.

2.9 JavaScript Object Notation (JSON)

A JSON file is a lightweight file to store the data in a simple data structure and object in JSON format, which is also the standard data exchange format. The JSON file is text-based, human-readable, "self-describing" and easy to understand we can edit it using a text editor also. While they can save lot of information using two structure either collection of name/value pair or array. The name/value pair structure is used to model a object because we know that object is the collection of attributes holding some values and array structure is used to model list. Every object in JSON is modelled using {..} and attribute can be modelled as name:value pair. Here value can me object, array, or simple primitive value and so on. We can parse the JSON data on Android, for this we need to create the instance of JSONObject and JSONArray objects with the strings containing the data on it [7].

Chapter 3

Design and Solution

3.1 Design

This section describes about the designing of the mTeleHealth-UiA app, based on the application framework design. It then presents some wireframe design of the application which has to be designed before actual coding or developing of the app begins, which is a professional technique to make a work lot easier and visualize the app in general as a whole. After that we will be looking out the design of database of our application which visualize how are data to be saved in database and make us easy afterwards for others in further development and also describes about the design for symptom assessment.

3.1.1 Framework Design

Usually, the health workers in remote areas have been trained with primary health care knowledge, which make their knowledge very limited for examining the patient at a very early stage. The early stage exam helps to treat the patient effectively and create a good report and overview of health status for further diagnosis. Traditionally they record patient vital signs, medical history, symptoms in the paper, which then is referred to the doctor along with patient visit. This may cause an error on recording patient details on paper. Also, there is a risk of not understanding the language or sentence written by a health worker by the doctor. The patient may also lose the report or misplace somewhere, and these factors may cause a delay in health service.

To solve this problem, we design the application, which overall framework is shown in figure 3.1. mTeleHealth app uses the mobile device with the Continua Certified PHD, which automatically transfers and saves all the vital signs data on health device to the mobile device. So, the health worker does not need to record every time they measure vital signs of the patient and write on paper.

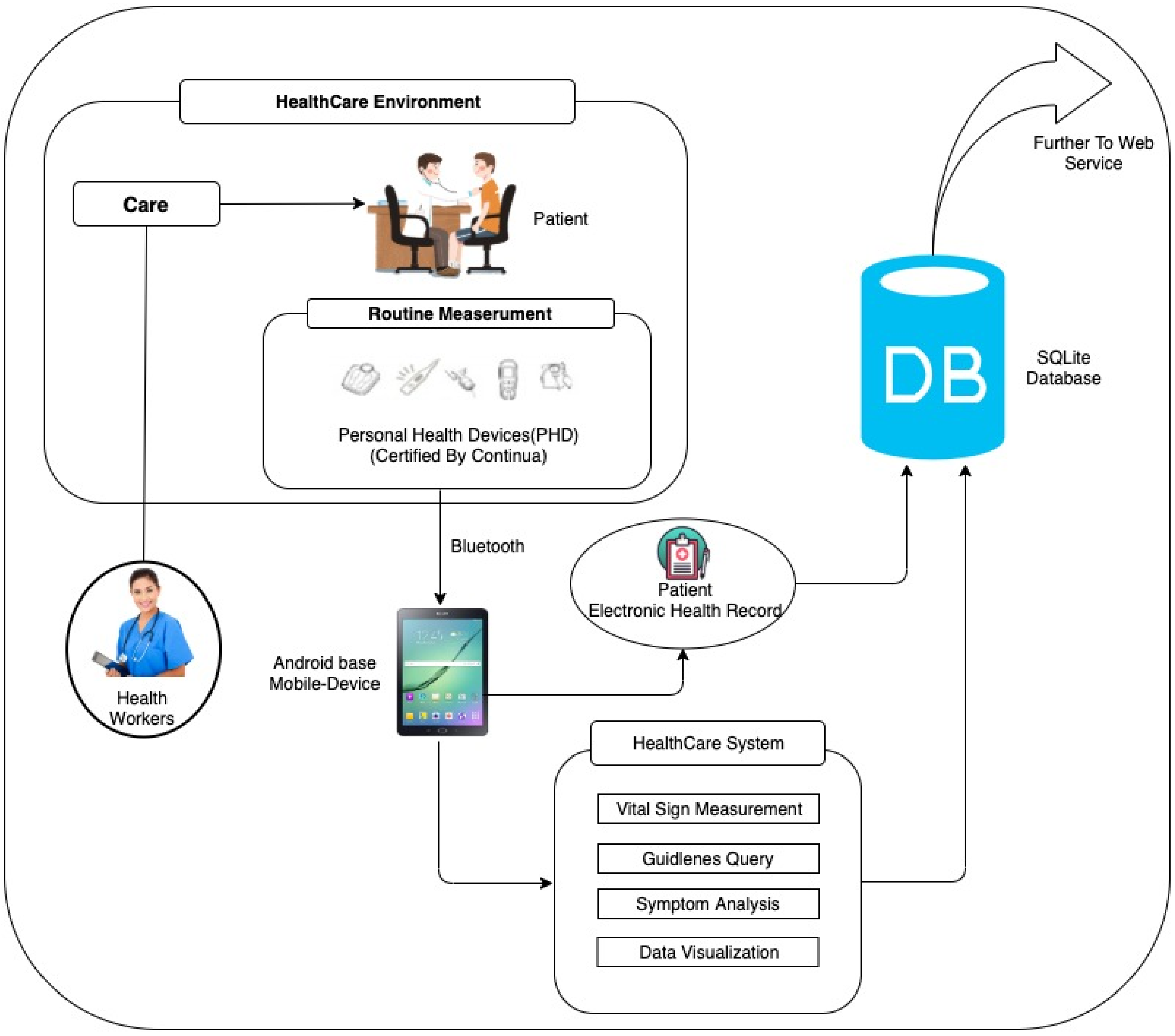


Figure 3.1: The framework of mTeleHealth-UiA

Furthermore, health workers rely on their basic health knowledge and experience to tell whether patient vital signs are normal or not without using any reference. This is inconvenient and can cause the severe condition and lower quality of patient care. For this purpose, we make a guideline based on the National Early Warning Score (NEWS) on the application for the health care system. After measuring the vital signs of the patient, the standardized guidelines are used for the assessment and response to acute illness. The guidelines calculate the total score and display it as NEWS score, which gives a clear picture of the acute status of the patient along with some care guidelines or information about the measured vital sign. These guidelines and score calculation details will be discussed in later section. mTeleHealth app is also designed to record the patient details, family and medical history details and save all the data into the database. Also, this app can analyze the information based on the user input and gives out the possible other similar symptoms, overview of symptoms, how common is this issue, how to treat, when to see a doctor, how is the risk factors, how it is made worse and is it emergency to see doctor or not.

For example, a scenario of the above framework is that in a health care environment a patient comes for health examine on the health post where health workers examine the patient and record all the data in Android-based Mobile device with the support of the mTelehealth app. Health worker creates a profile of the patient if the visit is new, else if he had already visited before health worker search for the patient name in the app. After registering or retrieving the patient details, health workers start a new visit of the patient and record his vital signs manually or automatically from the health devices. After measuring and saving the vital signs data with the help of app health worker will get an overview of the acute status of the patient, the app will provide NEWS score and some guidelines in real-time. Then the patient family and medical history are taken and examined with the questions, and the symptom assessment is being done and shown out the possible findings for an overview, and all the data are saved on the database of the device. The data which are saved on the device they are ready to send to web service or expert medical personnel for further assessment, which is not implemented yet and is future work.

An Android application mTeleHealth for mobile devices is developed to help the health worker and carry out the goals of our framework. The application provides major functions such as measurement of vital signs, provides a knowledge base of care guidelines, symptoms that are employed to trace the overview of health status, saving patient health records and make ready to be sent further.

3.1.2 Application Wireframe

A wireframe is a page schematic or screen blueprint, that can represent the visual and skeletal framework, for a mobile application it is a two-dimensional representation of the mobile screen interface. It is also the visual concept of the application, which is in the process of development in the future. It provides design and overviews how the application will work. Below in figure 3.2, we can see the part of the wireframe of our mTeleHealth application, which will be developed accordingly.

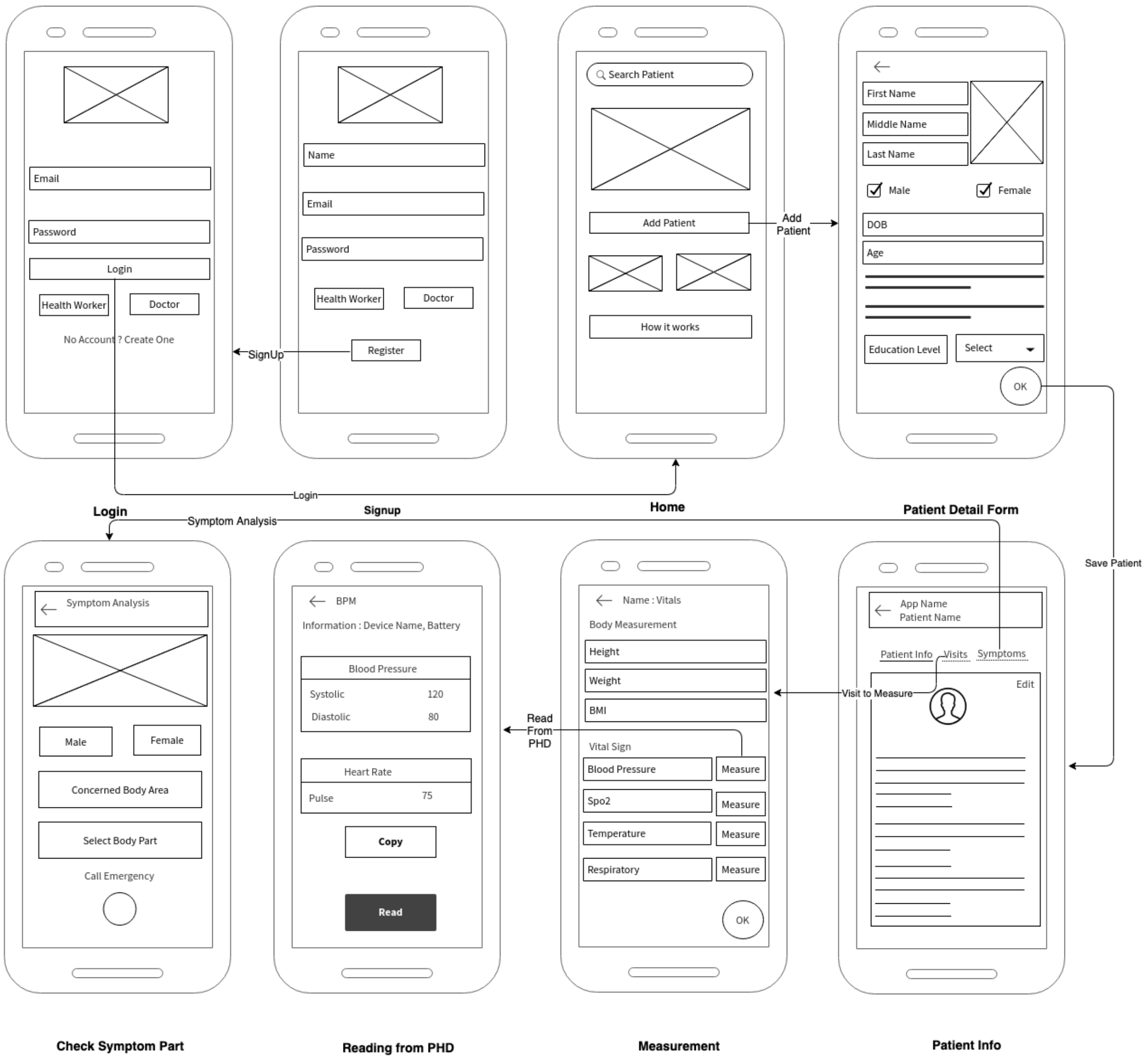


Figure 3.2: The Wireframe for mTeleHealth-UiA

These wireframes are developed to match the requirement of the research project [26] project no AHRC (UK) Reference: AH/R003882/1. Above wireframe shows the page elements, object categories, content prioritization, possible actions, and visual branding element of the application screen interface. However, it does not show design elements, colors used, actual images, fonts, and logos which are going to be finalized in the implementation or coding phase. We can see above wireframing of an app where it shows a clear overview of the app, what are our expectations from the app and what functions do we need in our app to have. It helps to decide for changes before actual implementation starts, which quite easy; afterward, it is so costly for changes. We had also created wireframes for screen interface of login, signup, the home page, patient detail fill up the page, patient info showing page, measurement of vitals page, how to read and copy data from PHD page, symptoms questionnaire page and others. Also, the flow of app can be seen from wireframe, from above we know that when the login button is pressed, then it should display home screen and if add patient is clicked on home screen patient detail form should be displayed. Similarly, by wireframing, it can be easy to see the complete overview of the application and be easy in development phase.

3.1.3 Database Design

The database is an organized collection of the data saved and accessed in electronic form in memory storage. Where there is much information to save and retrieve then its complex to visualize and maintain the database, so it is essential to model our data about how it will be saved. The ER or (Entity Relational Model) is widely used for high-level conceptual modeling of data. It helps us to analyze data requirements systematically and to construct a well-designed database. It is also the relation of the entities stored in the database and explain the logical structure of databases. So, many professional considered it to be best practice before implementing the database. Hence, we also design the database diagram, as shown in figure 3.3. Here we can find three different term entities, attributes, and cardinality. The entity is a real-world thing which is to be represented in our database; for example, they can be a person, place, object, event, or concept. Attributes are the property of the entity such as a person entity can have employee, student, and patient as an attribute. Cardinality is nothing but the relationship between the entities. Here we design a database with eight entities which will be similar as the tables created on our database, and each got their attributes which will be saved in row and column in our database. Similarly, the database is modeled for our application.

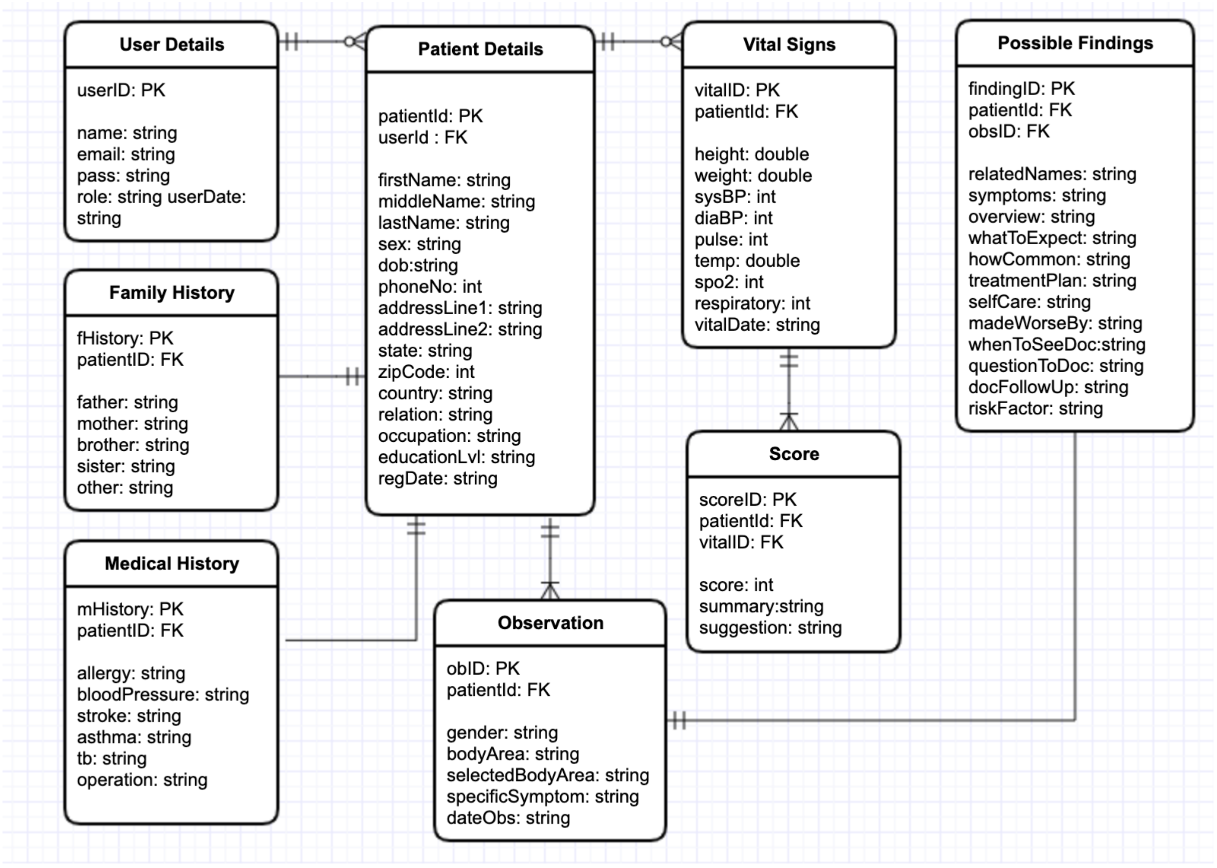


Figure 3.3: Modelling of Database

3.1.4 Design for Symptoms Assessment

There are numerous online application and tool for the symptom assessment, which gives the result according to the input answers and is quite beneficial for to get an overview of the medical condition before visiting the doctor. So we also want to build a symptom assessment tool in our application which can give an overview of the patient medical condition. As we know, our app is mainly focused on the rural areas where there may or may not be the availability of the network. So, the information provided by the application should be based locally without querying the web server. So, here we should design the symptom assessment tool in such a way that it can save all the knowledge base information locally and can be retrieved smoothly. One way to achieve this is to map the symptoms and answers provided by the patient and to reach a conclusion to display the summary or result as shown in figure 3.4

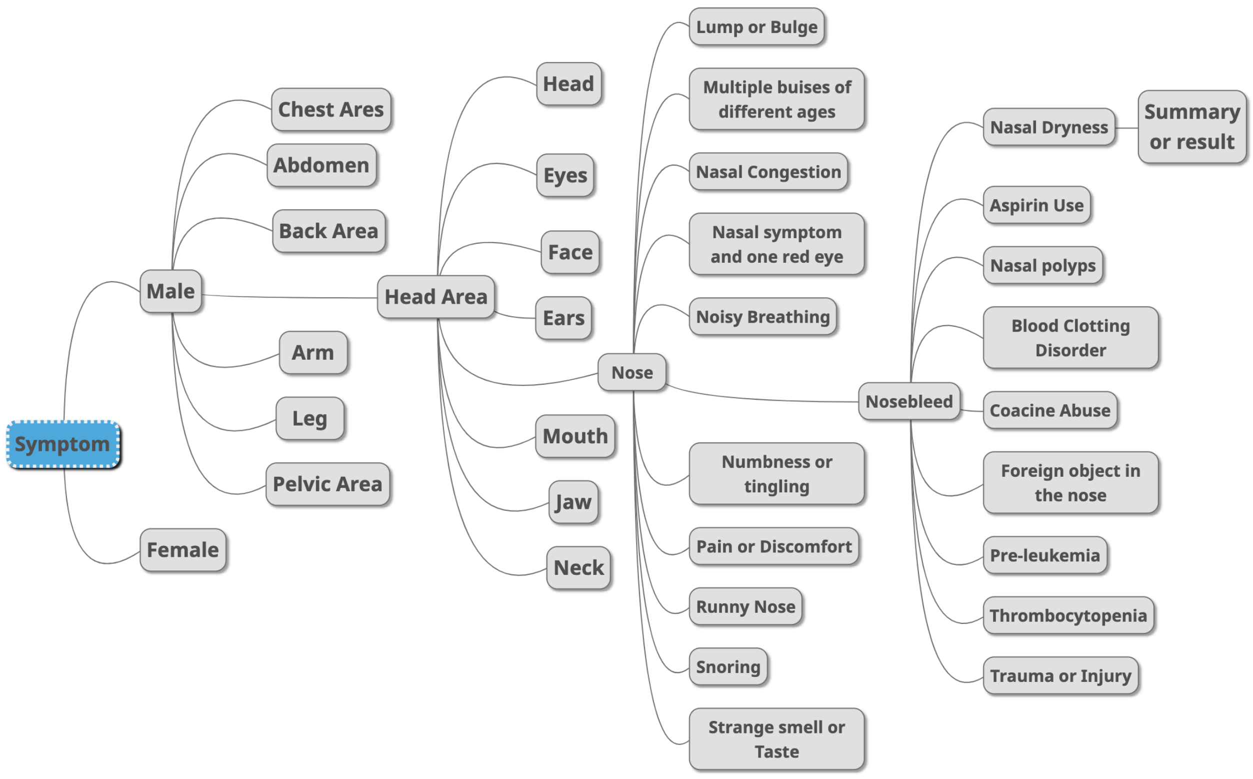


Figure 3.4: MindMap for symptom Assessment

Above shown diagram is a mind map technique for analyzing symptom, which helps us to organize information visually. Here we design in such a way that we should be able to get the summary or result of the patient health overview by doing the questionnaire session as below:

Q.What is patient gender?

A.Male.

Q.In which body area is patient feeling unwell?

A.Head.

Q.In which specific part of that body area patient is feeling unwell?

A.Nose.

Q.Which of the following specifically matches the patient symptom?

A.Nasal Dryness.

Now we will be able to get the summary or result of the patient medical status overview.

3.2 Solution

This section includes the solution of the important task of this application which includes the integration of the medical sensor in the app which uses Bluetooth and BLE for information exchange, calculating the score from the vital life signs and giving a recommendation, saving the information values in the database, and the process how symptoms are analyzed. Below table shows the tools we used for developing this application. Here we discuss about some of the important code snippet and the rest full code are submitted along with the final zip file with application package.

. . Programming Languages . .	Java XML (Extensible Markup Language) JSON (JavaScript Object Notation)
. Integrated Development Environment (IDE) .	Android Studio version 3.3
. Database .	SQLite
. Compatible with .	API Level: 21 Android Version smartphones from Lollipop (5.0) and above.

Table 3.1: Details of tools used

3.2.1 Overview of the App

In Android Studio, we created the new project for phone and tablet and chose an empty activity and named the project as mTeleHealth-UiA. The figure 3.5 shows the complete solution and features of the application to be made. Here in the illustration, we can see that if the user is already registered, then they can Login else they can create profile and login and can reach the home page. Here they can register or search the patient is already registered. If we add patient, then the form will be opened where we save information of the patient, and by searching patient which are already saved, we get patient information and their visit tab where we can see their visit date. If the patient had previously visited before we can see their visit summary else, we could start a new visit. When we start a new visit, a Vitals measurement form will be opened where we fill all the vitals parameter. We can manually enter the parameters else we can record from the medical sensors automatically. After that, family and medical history are taken and saved for the summary. In the summary page, we can see an overview of the recorded vitals, a calculated score, and recommendation based on score, family and medical history. In summary, we also have a symptom assessment tool which can be used to analyze the medical condition of patient and application gives some recommendation.

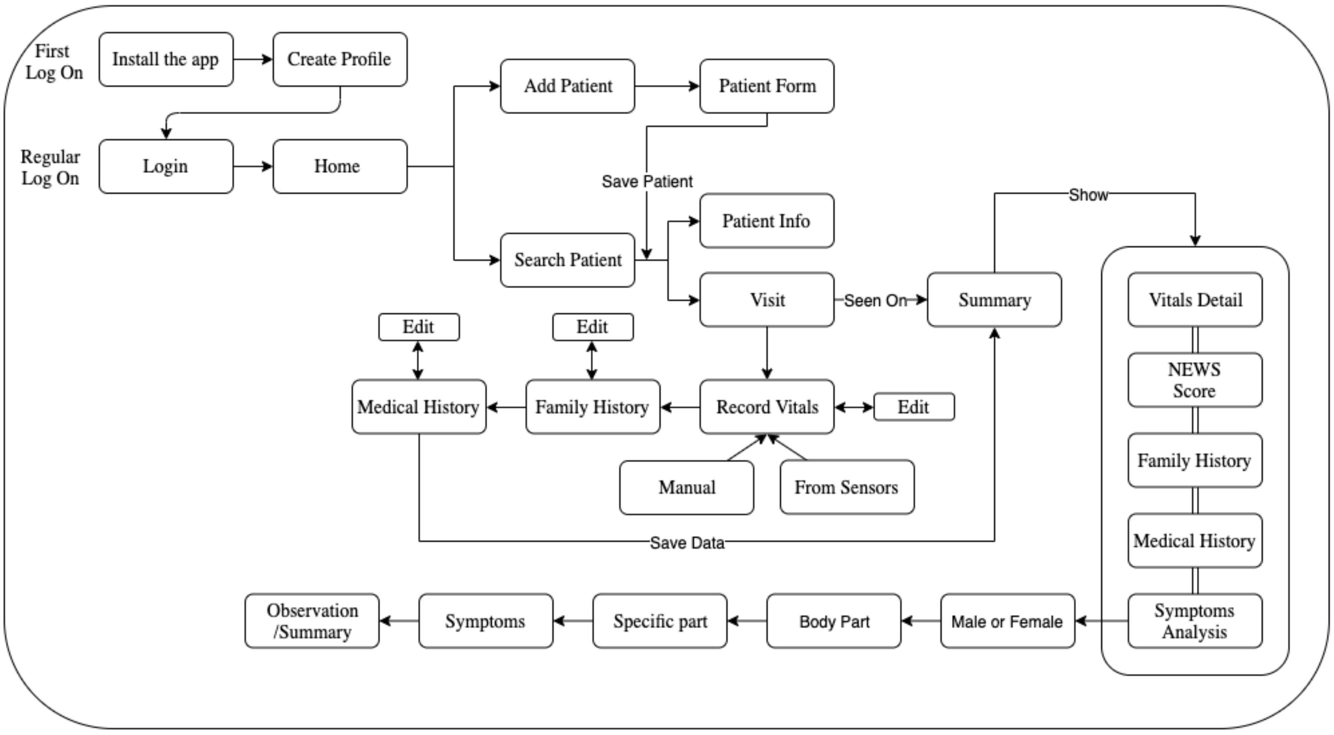


Figure 3.5: Simple representation of the features in the app

3.2.2 Reading data from PHD with BLE

This is one of the major difficult task faced during the development of this application. Although PHD uses a different form of technology for data transportation here, we use BLE to obtain data from our medical sensors (PHD) to gateway or device (Tablet). Here we need to develop an application for Android device which then can communicate with the PHD which are also equipped with BLE. There are a variety of health devices such as blood pressure monitor, thermometer, heart-rate monitor each of these has different profile its services, characteristics, UUID and descriptors. Hence, to read data from these health devices, we need to create an application based on each profile. In this project, we use two health devices which are equipped with the BLE, Health Thermometer and Blood Pressure Monitor. The process of building the API is similar for all the health devices; there will only be a change in service, characteristics value and descriptor value. So, here we will be creating an application for Health Thermometer for our Android device which then will be able to read the measured data from the Health Thermometer device.

Our aim is to connect mTeleHealth-UiA app with Health Thermometer to read the measurements. Below shows the specification of the service and characteristic of the Health Thermometer which is standard and defined by the SIG group.

Health Thermometer Service

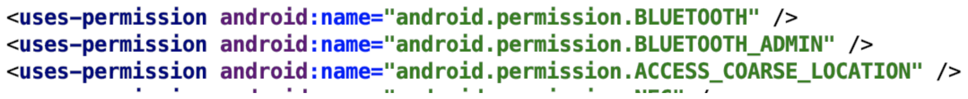
- Assigned Number/UUID: 0x1809
- Type: org.bluetooth.service.health_thermometer
- Definition: This service exposes temperature and other data from a Health Thermometer Sensor intended for fitness applications.

Temperature Measurement Characteristic:

- Assigned Number/UUID: 0x2A1C
- Type: org.bluetooth.characteristic.temperature_measurement
- Definition: This characteristic is used to send a temperature measurement.

- Properties: Indicate
- Descriptor: Client Characteristic Configuration

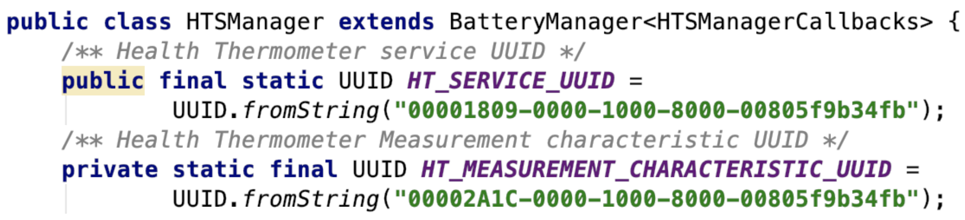
By looking at the specification, we know that in order to read the measurement we need to set notification properties on Temperature Measurement Characteristic to get data streaming. So, now we start to code by creating an empty activity and declare the BLE permission on Manifest file. Here we need to request location permission also due to LE beacon are always associated with the location to get scanning result as in figure 3.6.



```
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
```

Figure 3.6: Declare BLE Permissions

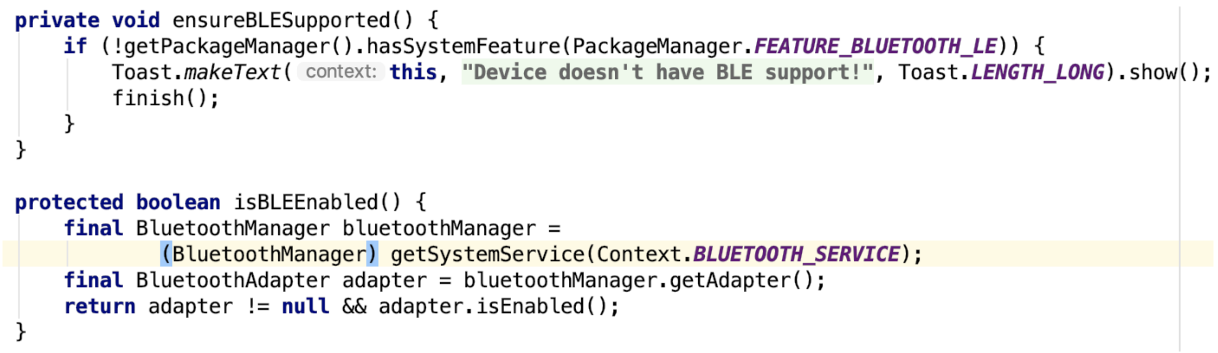
Now assigning UUID for service and characteristic as in figure 3.7



```
public class HTSManager extends BatteryManager<HTSManagerCallbacks> {
    /** Health Thermometer service UUID */
    public final static UUID HT_SERVICE_UUID =
            UUID.fromString("00001809-0000-1000-8000-00805f9b34fb");
    /** Health Thermometer Measurement characteristic UUID */
    private static final UUID HT_MEASUREMENT_CHARACTERISTIC_UUID =
            UUID.fromString("00002A1C-0000-1000-8000-00805f9b34fb");
```

Figure 3.7: Assigning UUID

Setting BLE in our activity and checking weather our device support BLE or not.

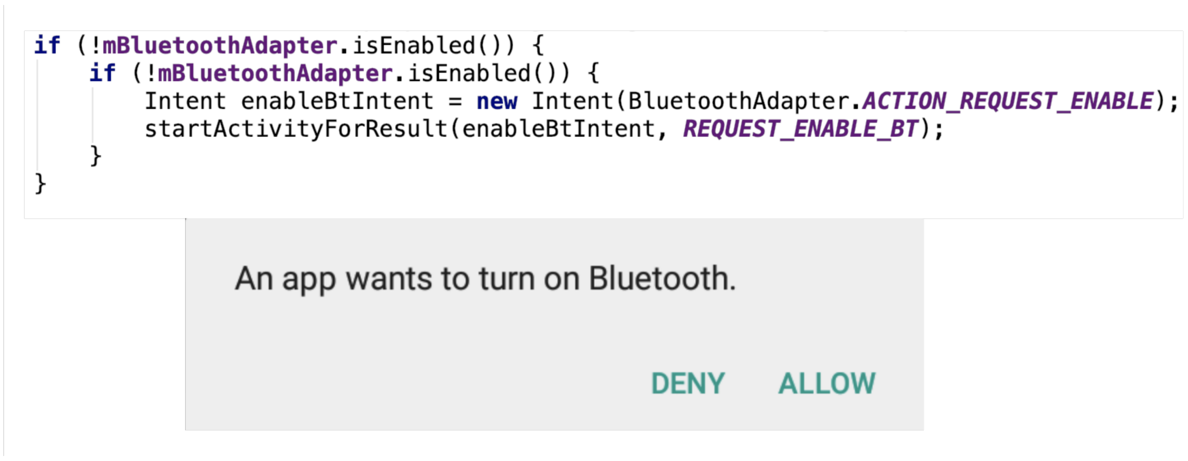


```
private void ensureBLESupported() {
    if (!getPackageManager().hasSystemFeature(PackageManager.FEATURE_BLUETOOTH_LE)) {
        Toast.makeText( context: this, "Device doesn't have BLE support!", Toast.LENGTH_LONG).show();
        finish();
    }
}

protected boolean isBLEEnabled() {
    final BluetoothManager bluetoothManager =
            (BluetoothManager) getSystemService(Context.BLUETOOTH_SERVICE);
    final BluetoothAdapter adapter = bluetoothManager.getAdapter();
    return adapter != null && adapter.isEnabled();
}
```

Figure 3.8: Set BLE and Checking its availability on device

Now after knowing that the BLE is on the device ensure that it is enabled for scanning if not fire an intent and display a dialogue box asking the permission from user as below figure 3.9.

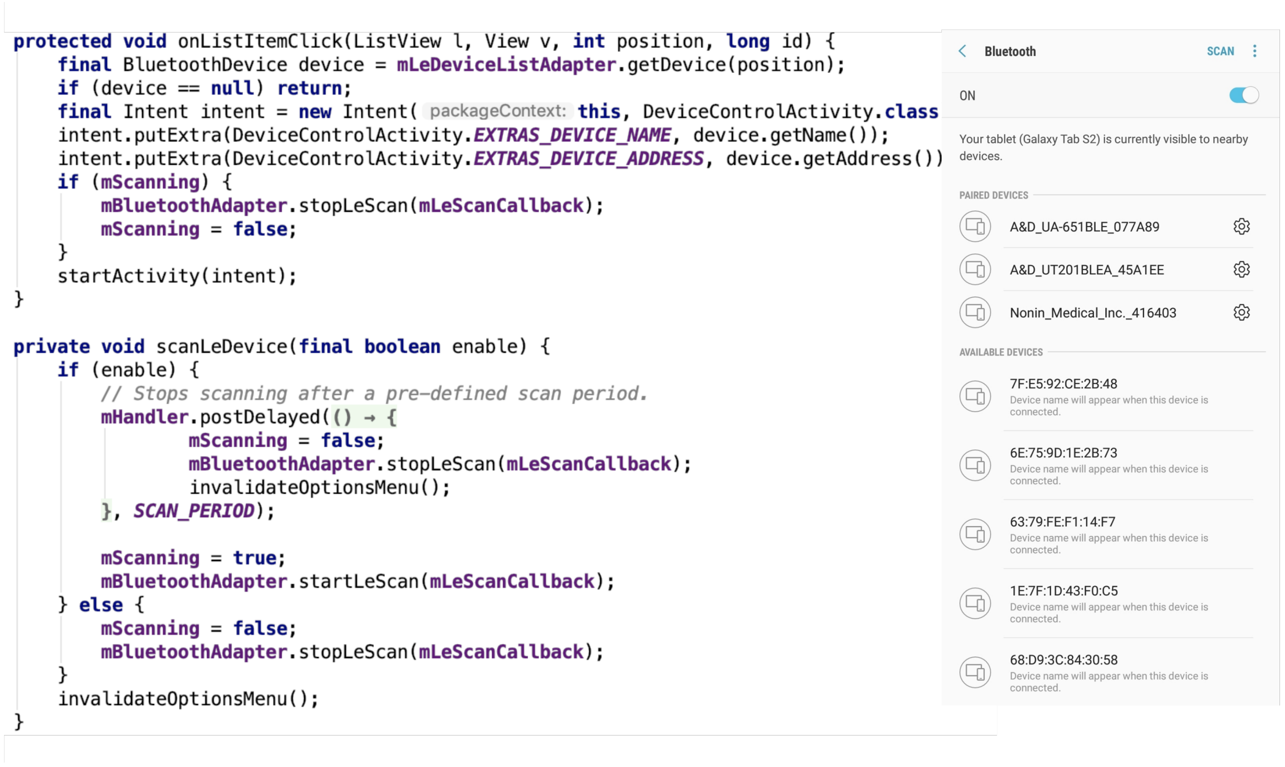


```
if (!mBluetoothAdapter.isEnabled()) {
    if (!mBluetoothAdapter.isEnabled()) {
        Intent enableBtIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
        startActivityForResult(enableBtIntent, REQUEST_ENABLE_BT);
    }
}
```

An app wants to turn on Bluetooth.

Figure 3.9: Event to turn On the Bluetooth & dialogue on device

After enabling the Bluetooth scan for devices nearby and display the device address and names as in figure 3.10.



```
protected void onListItemClick(ListView l, View v, int position, long id) {
    final BluetoothDevice device = mLeDeviceListAdapter.getDevice(position);
    if (device == null) return;
    final Intent intent = new Intent( packageContext: this, DeviceControlActivity.class
    intent.putExtra(DeviceControlActivity.EXTRAS_DEVICE_NAME, device.getName());
    intent.putExtra(DeviceControlActivity.EXTRAS_DEVICE_ADDRESS, device.getAddress())
    if (mScanning) {
        mBluetoothAdapter.stopLeScan(mLeScanCallback);
        mScanning = false;
    }
    startActivity(intent);
}

private void scanLeDevice(final boolean enable) {
    if (enable) {
        // Stops scanning after a pre-defined scan period.
        mHandler.postDelayed(() → {
                mScanning = false;
                mBluetoothAdapter.stopLeScan(mLeScanCallback);
                invalidateOptionsMenu();
        }, SCAN_PERIOD);

        mScanning = true;
        mBluetoothAdapter.startLeScan(mLeScanCallback);
    } else {
        mScanning = false;
        mBluetoothAdapter.stopLeScan(mLeScanCallback);
    }
    invalidateOptionsMenu();
}
```

ON

Your tablet (Galaxy Tab S2) is currently visible to nearby devices.

PAIRED DEVICES

A&D_UA-651BLE_077A89

A&D_UT201BLEA_45A1EE

Nonin_Medical_Inc._416403

AVAILABLE DEVICES

7F:E5:92:CE:2B:48
Device name will appear when this device is connected.

6E:75:9D:1E:2B:73
Device name will appear when this device is connected.

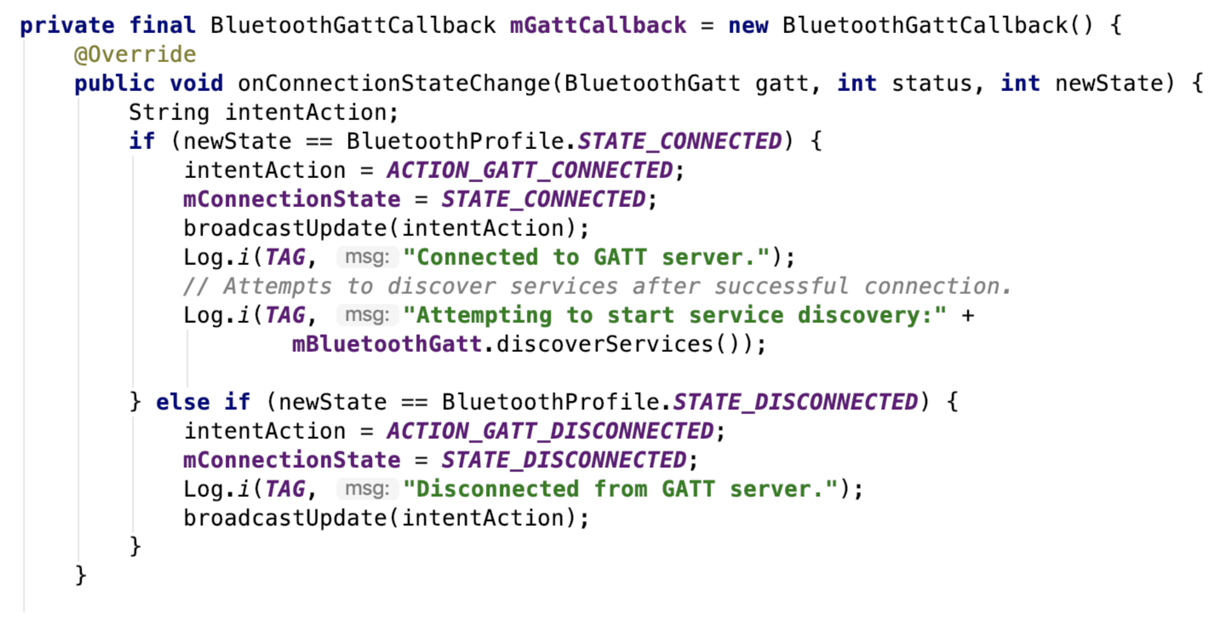
63:79:FE:F1:14:F7
Device name will appear when this device is connected.

1E:7F:1D:43:F0:C5
Device name will appear when this device is connected.

68:D9:3C:84:30:58
Device name will appear when this device is connected.

Figure 3.10: Scanning Nearby Devices & Dialogue on device

After discovering the devices we need to connect to the GATT server on health device for managing connection and data communication. So to connect we define GATT Client & callback.

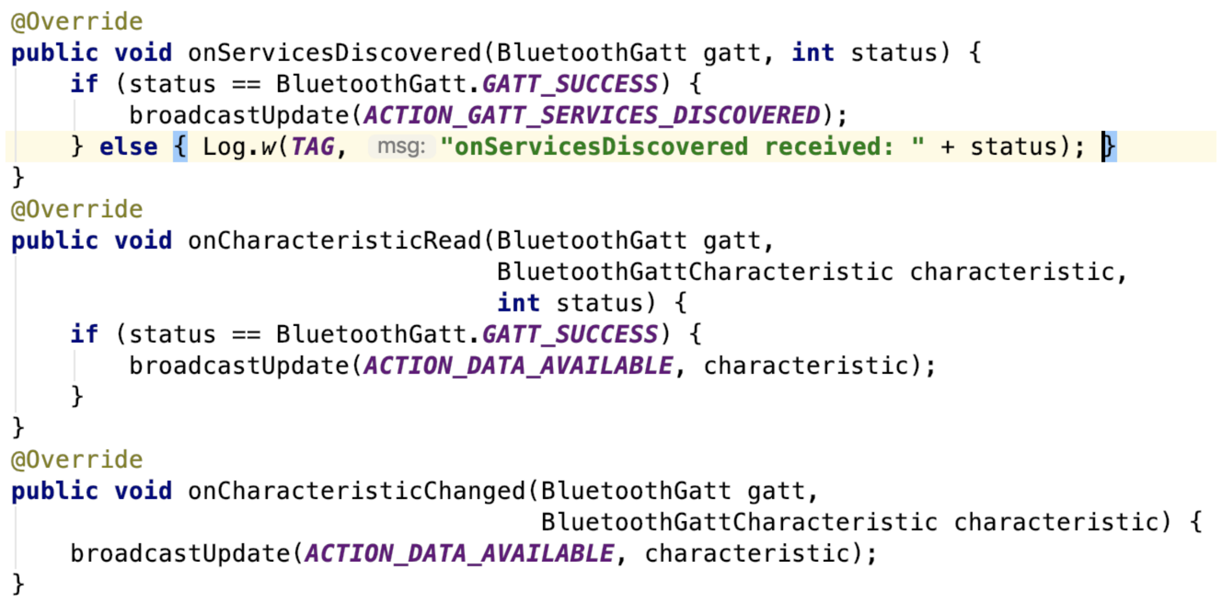


```
private final BluetoothGattCallback mGattCallback = new BluetoothGattCallback() {
    @Override
    public void onConnectionStateChange(BluetoothGatt gatt, int status, int newState) {
        String intentAction;
        if (newState == BluetoothProfile.STATE_CONNECTED) {
            intentAction = ACTION_GATT_CONNECTED;
            mConnectionState = STATE_CONNECTED;
            broadcastUpdate(intentAction);
            Log.i(TAG,  msg: "Connected to GATT server.");
            // Attempts to discover services after successful connection.
            Log.i(TAG,  msg: "Attempting to start service discovery:" +
                    mBluetoothGatt.discoverServices());

        } else if (newState == BluetoothProfile.STATE_DISCONNECTED) {
            intentAction = ACTION_GATT_DISCONNECTED;
            mConnectionState = STATE_DISCONNECTED;
            Log.i(TAG,  msg: "Disconnected from GATT server.");
            broadcastUpdate(intentAction);
        }
    }
```

Figure 3.11: Connecting to GATT server.

When the pairing or connection is established then we start to discover the provided services.

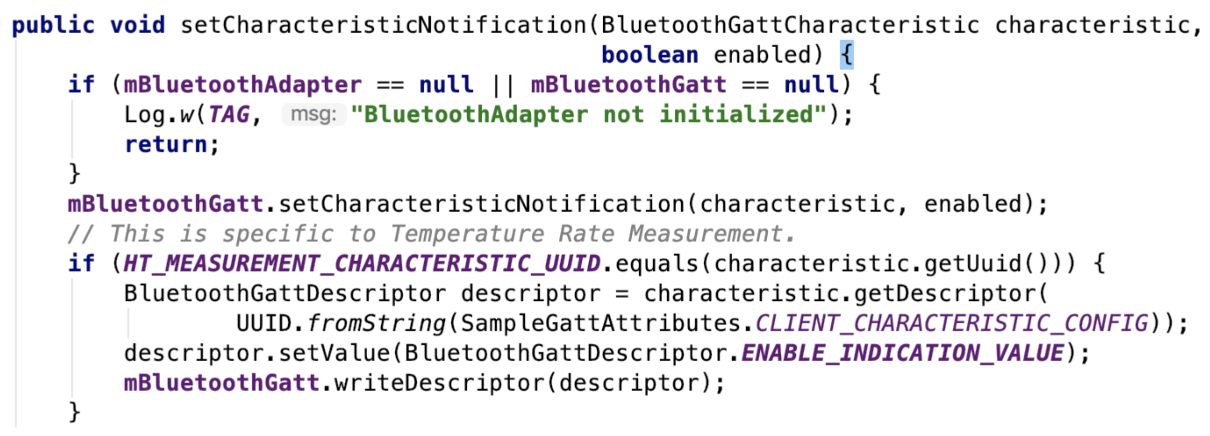


```
@Override
public void onServicesDiscovered(BluetoothGatt gatt, int status) {
    if (status == BluetoothGatt.GATT_SUCCESS) {
        broadcastUpdate(ACTION_GATT_SERVICES_DISCOVERED);
    } else { Log.w(TAG,  msg: "onServicesDiscovered received: " + status); }
}
@Override
public void onCharacteristicRead(BluetoothGatt gatt,
                                 BluetoothGattCharacteristic characteristic,
                                 int status) {
    if (status == BluetoothGatt.GATT_SUCCESS) {
        broadcastUpdate(ACTION_DATA_AVAILABLE, characteristic);
    }
}
@Override
public void onCharacteristicChanged(BluetoothGatt gatt,
                                    BluetoothGattCharacteristic characteristic) {
    broadcastUpdate(ACTION_DATA_AVAILABLE, characteristic);
}
```

Figure 3.12: Discovering and reading the service and characteristic.

Now our aim is to read the temperature value so we should set notification or indication on Temperature Measurement Characteristic, so to set the notification or indication value we tell the sensor to enable us this notification or indication mode. We should then write to the descriptor to set right value for characteristics.

Now the updates from the sensors on characteristics value will be posted on the next call



```java
public void setCharacteristicNotification(BluetoothGattCharacteristic characteristic,
                                          boolean enabled) {
    if (mBluetoothAdapter == null || mBluetoothGatt == null) {
        Log.w(TAG,  msg: "BluetoothAdapter not initialized");
        return;
    }
    mBluetoothGatt.setCharacteristicNotification(characteristic, enabled);
    // This is specific to Temperature Rate Measurement.
    if (HT_MEASUREMENT_CHARACTERISTIC_UUID.equals(characteristic.getUuid())) {
        BluetoothGattDescriptor descriptor = characteristic.getDescriptor(
                UUID.fromString(SampleGattAttributes.CLIENT_CHARACTERISTIC_CONFIG));
        descriptor.setValue(BluetoothGattDescriptor.ENABLE_INDICATION_VALUE);
        mBluetoothGatt.writeDescriptor(descriptor);
    }
```

Figure 3.13: Reading the value from the characteristic.

back using onCharacteristicChanged from above. And finally we will get the temperature measurement data from health thermometer.

In our application, we implement the UI as below figure 3.14 which shows the process to read temperature measurement. In the figure, we can see five different screen interface and one image of health thermometer reading. The screen interface 1 illustrates the form to record different vital parameter as above discussed, in this screen interface when we press measure it opens the screen 2 where we get the data from the health thermometer. Before pressing the read button from screen 2, we should measure the temperature of a patient by the health thermometer, which is shown in image 6 from the figure. After the measurement the device broadcast the data through BLE, and the receiving device should connect to the thermometer through the GATT server. When the read button is pressed from screen 2 and thermometer, broadcast the data the name of the thermometer is shown in mobile device list which is nearby as in screen 3. After that, when we click on the thermometer name GATT server is created, and the services are discovered, and data is read from the measurement characteristic which process is discussed above already. After reading the data, it is displayed as in the screen 4 and pressing ok it is saved in the vital measurement form under temperature section which then further saved on a database for further analysis. In the similar way, we can read values from Blood Pressure Monitor and other PHD which also support BLE and save the data values.

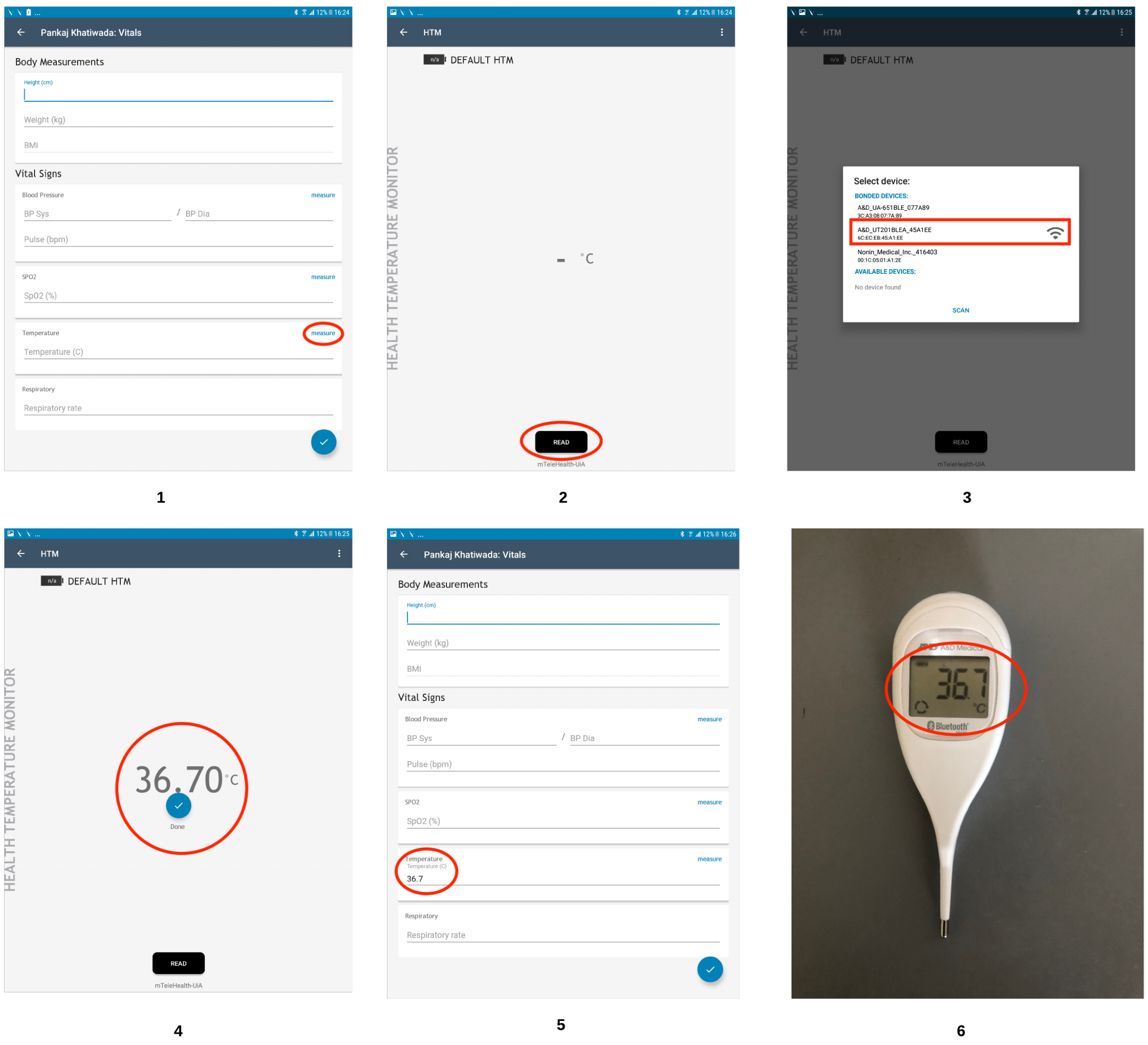
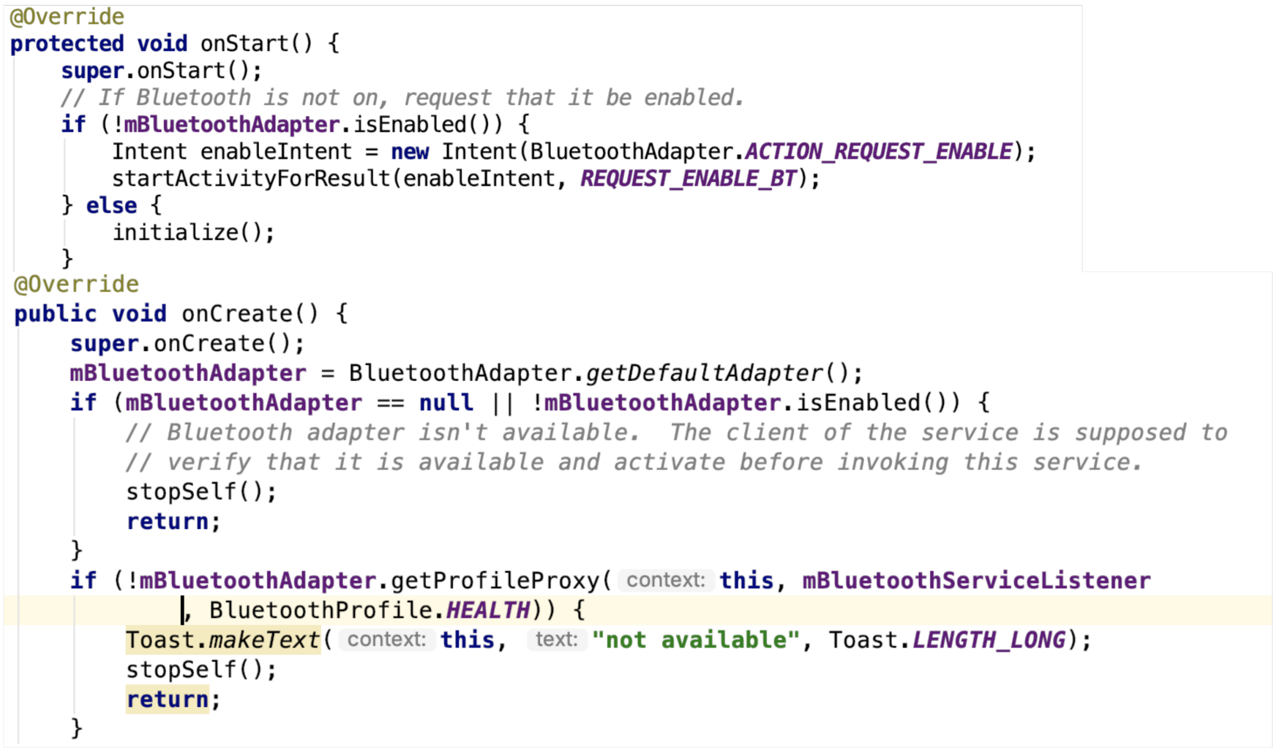


Figure 3.14: Process of Measuring temperature from mTeleHealth app.

3.2.3 Reading data from PHD with Classic Bluetooth

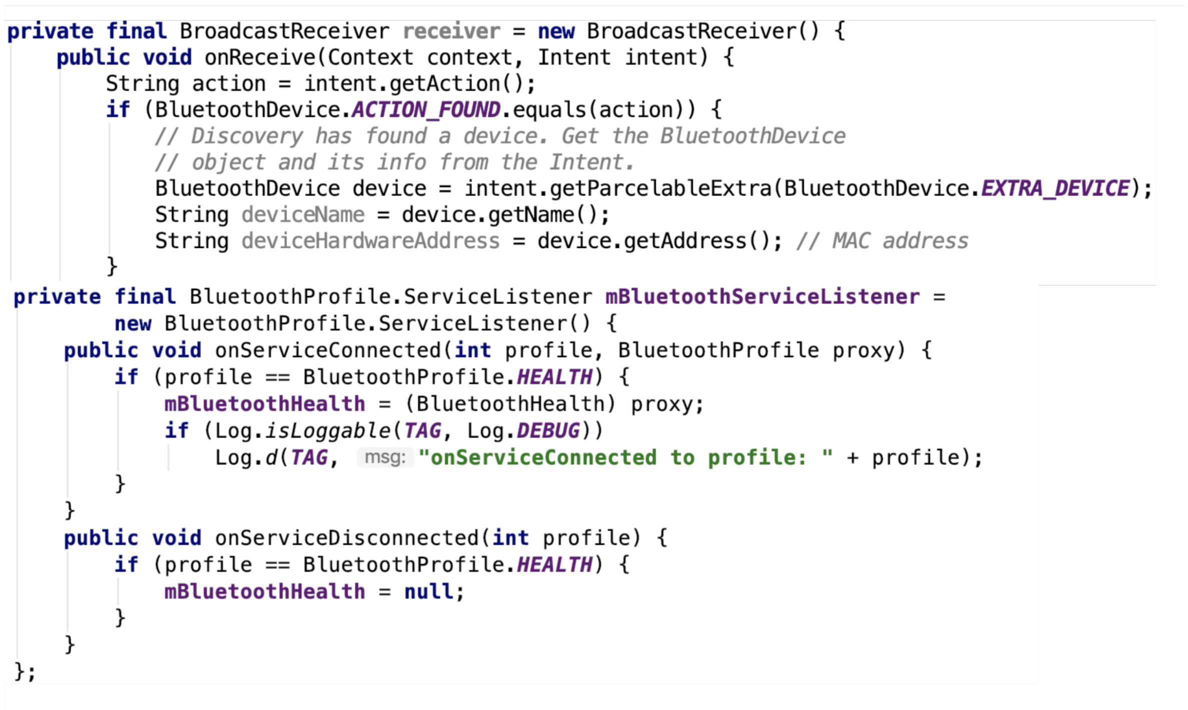
Reading the data from classic Bluetooth is somehow similar to the BLE scan, connect, and exchange information. But instead of GATT profile, it uses Bluetooth Health Device Profile (HDP) which is supported by Android. This profile lets us create an application that uses Bluetooth to communicate with the PHD. Here we use Spo2 health device which uses classic Bluetooth for exchanging blood oxygen measurement with our app. Here our aim is to read oxygen saturation on the patient from oximeter. So we should use the appropriate IEEE 11073 data exchange specifications types based on the devices used approved by the Bluetooth SIG group. So, the data type for pulse oximeter is 0x1004. Now we start to code by creating empty activity and services to declare permission similar as BLE in figure 3.6 above.



```
@Override
protected void onStart() {
    super.onStart();
    // If Bluetooth is not on, request that it be enabled.
    if (!mBluetoothAdapter.isEnabled()) {
        Intent enableIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
        startActivityForResult(enableIntent, REQUEST_ENABLE_BT);
    } else {
        initialize();
    }
@Override
public void onCreate() {
    super.onCreate();
    mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
    if (mBluetoothAdapter == null || !mBluetoothAdapter.isEnabled()) {
        // Bluetooth adapter isn't available.  The client of the service is supposed to
        // verify that it is available and activate before invoking this service.
        stopSelf();
        return;
    }
    if (!mBluetoothAdapter.getProfileProxy( context: this, mBluetoothServiceListener
            , BluetoothProfile.HEALTH)) {
        Toast.makeText( context: this,  text: "not available", Toast.LENGTH_LONG);
        stopSelf();
        return;
    }
```

Figure 3.15: Ensuring Bluetooth and HDP availability

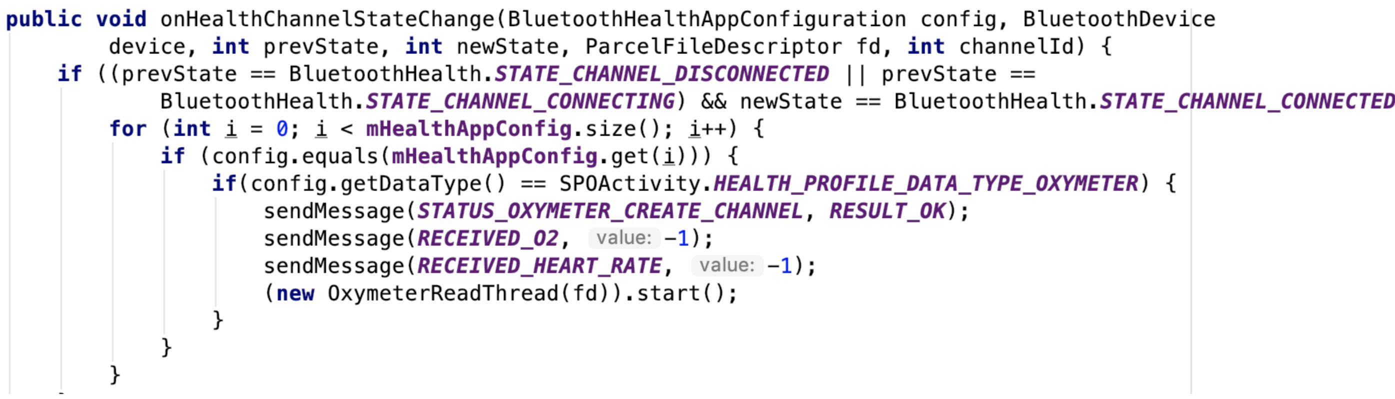
Before connecting to Bluetooth we should ensure availability of Bluetooth and health profile are available on device and enabled using the code as in figure 3.15. If success then we should start scanning for the available devices nearby using the code as in figure 3.16 and the list of devices are shown as in figure 3.9. Then we start to connect the gateway device and medical device using the Bluetooth HDP profile.



```
private final BroadcastReceiver receiver = new BroadcastReceiver() {
    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();
        if (BluetoothDevice.ACTION_FOUND.equals(action)) {
            // Discovery has found a device. Get the BluetoothDevice
            // object and its info from the Intent.
            BluetoothDevice device = intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
            String deviceName = device.getName();
            String deviceHardwareAddress = device.getAddress(); // MAC address
        }
private final BluetoothProfile.ServiceListener mBluetoothServiceListener =
        new BluetoothProfile.ServiceListener() {
    public void onServiceConnected(int profile, BluetoothProfile proxy) {
        if (profile == BluetoothProfile.HEALTH) {
            mBluetoothHealth = (BluetoothHealth) proxy;
            if (Log.isLoggable(TAG, Log.DEBUG))
                Log.d(TAG,  msg: "onServiceConnected to profile: " + profile);
        }
    }
    public void onServiceDisconnected(int profile) {
        if (profile == BluetoothProfile.HEALTH) {
            mBluetoothHealth = null;
        }
    }
};
```

Figure 3.16: Scanning for Bluetooth devices and Connecting

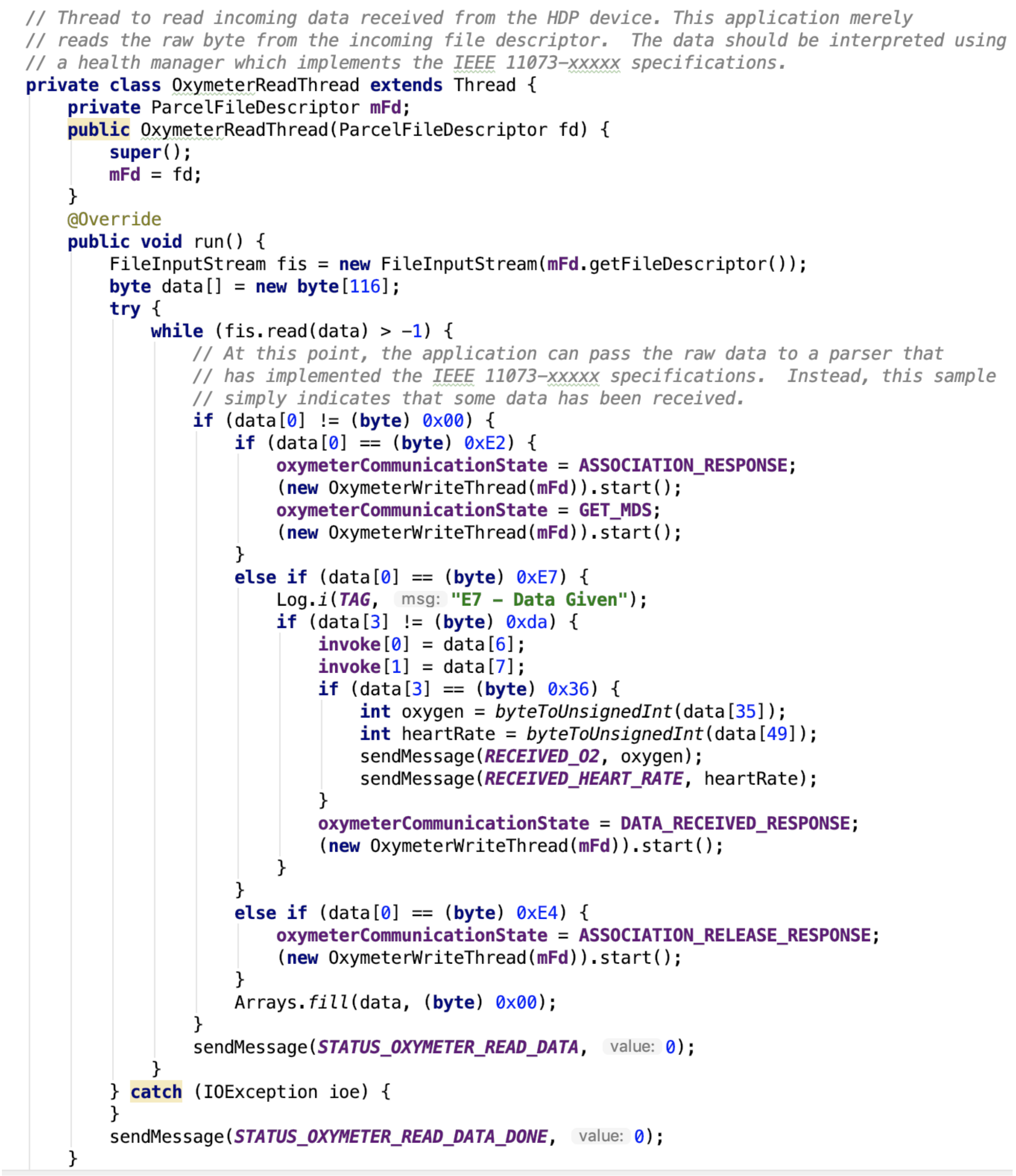
When the HDP device is connected then the gateway device start to read from the medical device using code as in figure 3.17 and the received file descriptor is passed to the OxymeterReadThread to read the content.



```
public void onHealthChannelStateChange(BluetoothHealthAppConfiguration config, BluetoothDevice
        device, int prevState, int newState, ParcelFileDescriptor fd, int channelId) {
    if ((prevState == BluetoothHealth.STATE_CHANNEL_DISCONNECTED || prevState ==
            BluetoothHealth.STATE_CHANNEL_CONNECTING) && newState == BluetoothHealth.STATE_CHANNEL_CONNECTED
        for (int i = 0; i < mHealthAppConfig.size(); i++) {
            if (config.equals(mHealthAppConfig.get(i))) {
                if(config.getDataType() == SPOActivity.HEALTH_PROFILE_DATA_TYPE_OXYMETER) {
                    sendMessage(STATUS_OXYMETER_CREATE_CHANNEL, RESULT_OK);
                    sendMessage(RECEIVED_O2,  value: -1);
                    sendMessage(RECEIVED_HEART_RATE,  value: -1);
                    (new OxymeterReadThread(fd)).start();
                }
            }
        }
```

Figure 3.17: Reading from HDP device

Below figure shows the code which is used to read the incoming data which is received by a HDP device and interpreted using IEEE 11073-XXXXX specifications.



```
// Thread to read incoming data received from the HDP device. This application merely
// reads the raw byte from the incoming file descriptor.  The data should be interpreted using
// a health manager which implements the IEEE 11073-xxxxx specifications.
private class OxymeterReadThread extends Thread {
    private ParcelFileDescriptor mFd;
    public OxymeterReadThread(ParcelFileDescriptor fd) {
        super();
        mFd = fd;
    }
    @Override
    public void run() {
        FileInputStream fis = new FileInputStream(mFd.getFileDescriptor());
        byte data[] = new byte[116];
        try {
            while (fis.read(data) > -1) {
                // At this point, the application can pass the raw data to a parser that
                // has implemented the IEEE 11073-xxxxx specifications.  Instead, this sample
                // simply indicates that some data has been received.
                if (data[0] != (byte) 0x00) {
                    if (data[0] == (byte) 0xE2) {
                        oxymeterCommunicationState = ASSOCIATION_RESPONSE;
                        (new OxymeterWriteThread(mFd)).start();
                        oxymeterCommunicationState = GET_MDS;
                        (new OxymeterWriteThread(mFd)).start();
                    }
                    else if (data[0] == (byte) 0xE7) {
                        Log.i(TAG, msg: "E7 - Data Given");
                        if (data[3] != (byte) 0xda) {
                            invoke[0] = data[6];
                            invoke[1] = data[7];
                            if (data[3] == (byte) 0x36) {
                                int oxygen = byteToUnsignedInt(data[35]);
                                int heartRate = byteToUnsignedInt(data[49]);
                                sendMessage(RECEIVED_O2, oxygen);
                                sendMessage(RECEIVED_HEART_RATE, heartRate);
                            }
                            oxymeterCommunicationState = DATA_RECEIVED_RESPONSE;
                            (new OxymeterWriteThread(mFd)).start();
                        }
                    }
                    else if (data[0] == (byte) 0xE4) {
                        oxymeterCommunicationState = ASSOCIATION_RELEASE_RESPONSE;
                        (new OxymeterWriteThread(mFd)).start();
                    }
                    Arrays.fill(data, (byte) 0x00);
                }
                sendMessage(STATUS_OXYMETER_READ_DATA,  value: 0);
            }
        } catch (IOException ioe) {
        }
        sendMessage(STATUS_OXYMETER_READ_DATA_DONE,  value: 0);
    }
```

Figure 3.18: Reading Incoming data from SPo2 device

In our application, we implement the UI as shown in below figure 3.19, which shows the process to read SPo2 measurement. In the figure, we see four different screen interface and one image of Oximeter reading SPo2 value. The screen interface 1 shows the form to record different vital parameter as discussed above already, in this screen interface when we press to measure; it opens the screen 2 where we can get the data from the SPo2 device. We made an automatic connection with the spo2 device before it is connected it shows red light as in screen interface 2 after connection it color changes to green and the reading are read from the device as of image 5 and displayed in the screen. After reading the data, it is displayed in the screen interface 3 and pressing the done button it is saved in the vital measurement form under SPo2 section shown in screen interface 4 which then further saved on a database for further analysis. All the UI code for the interfaces are attached on the appendix section below.

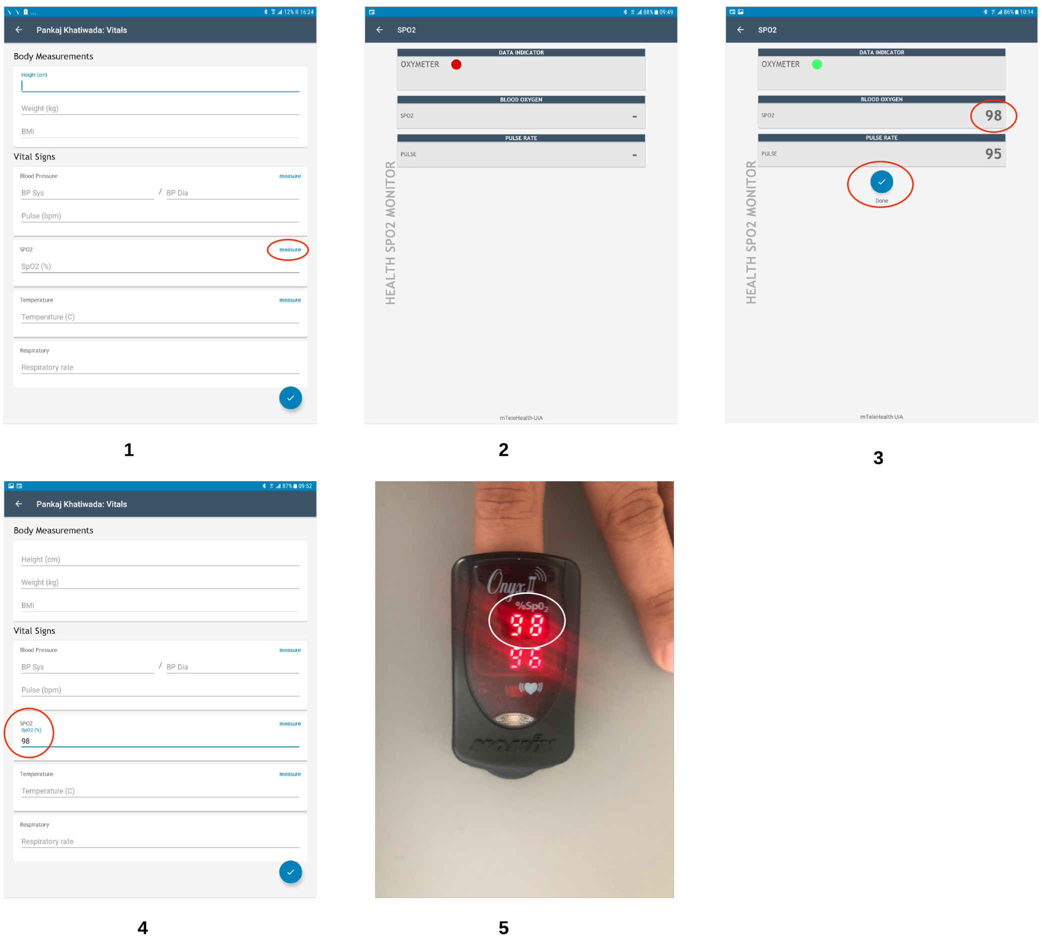
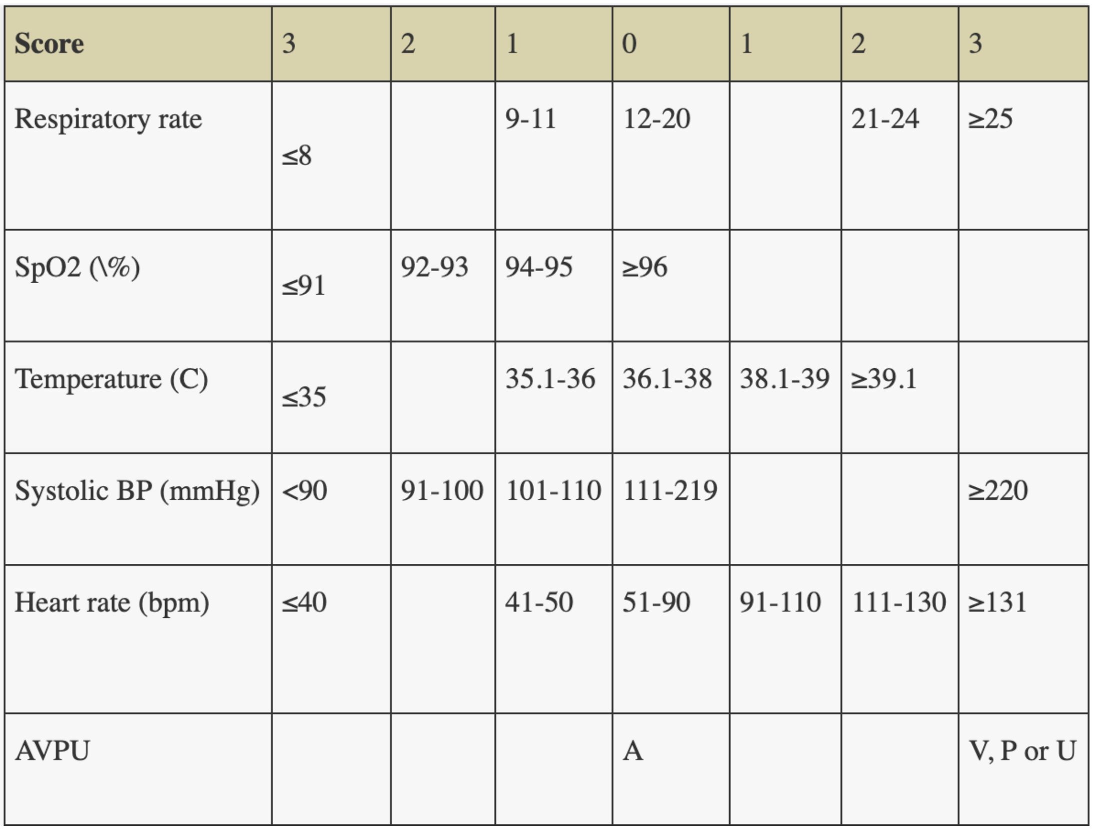


Figure 3.19: Process of Measuring SPo2 in mTeleHealth app

3.2.4 Calculating Early Warning Score

An early warning score is a medical guide used in the medical service, which determines the degree of the illness of the patient and instantly prompts for critical care intervention. It is based on the score value which is calculated by using vital signs values oxygen saturation, respiratory rate, pulse/heart rate, temperature, blood pressure and AVPU(Alert, Verbal, Pain, Unresponsive) response [38]. The NEWS can be applied to all the patient who is being hospitalized or the patient who came up for the regular checkup which allows the early detection of the clinical deterioration and provides information for the possible care intervention for higher level cases. The observation of all the vital parameter is compared to the normal range to generate a single composite score, for instance, based on the table 3.2, as shown below.

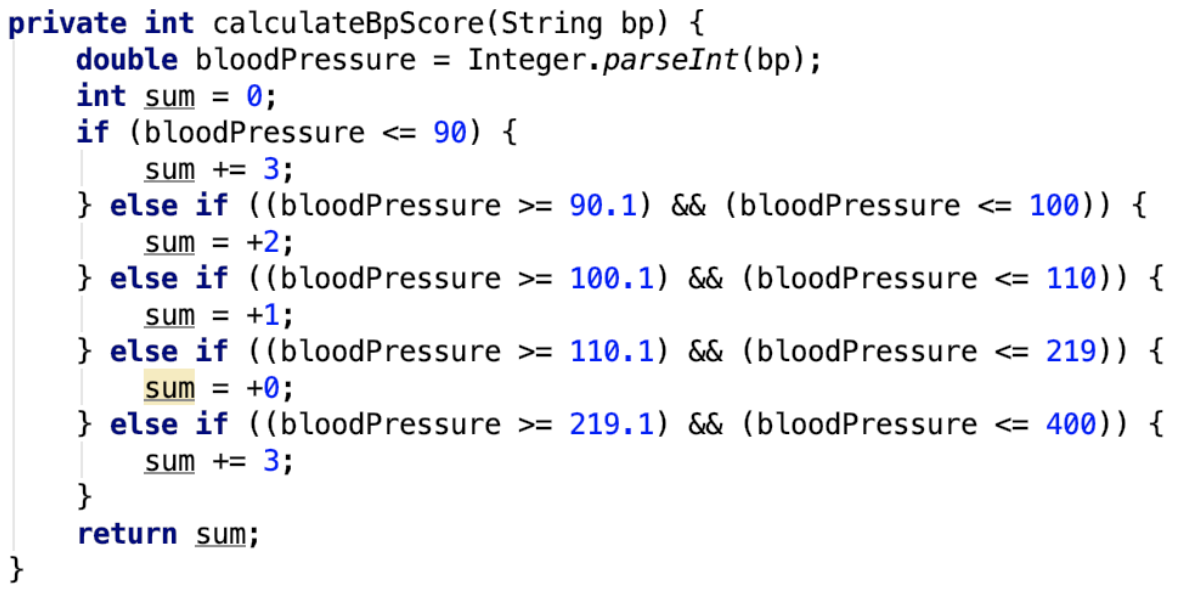


Score	3	2	1	0	1	2	3
Respiratory rate	≤8		9-11	12-20		21-24	≥25
SpO2 (\%)	≤91	92-93	94-95	≥96			
Temperature (C)	≤35		35.1-36	36.1-38	38.1-39	≥39.1	
Systolic BP (mmHg)	<90	91-100	101-110	111-219			≥220
Heart rate (bpm)	≤40		41-50	51-90	91-110	111-130	≥131
AVPU				A			V, P or U

Table 3.2: NEWS Score Calculation Table

This NEWS was the modified form of the Early Warning Score(EWS) and developed in the United Kingdom for the standard approach to identify the clinical deterioration in acutely ill patients and had bee widely adopted [21].

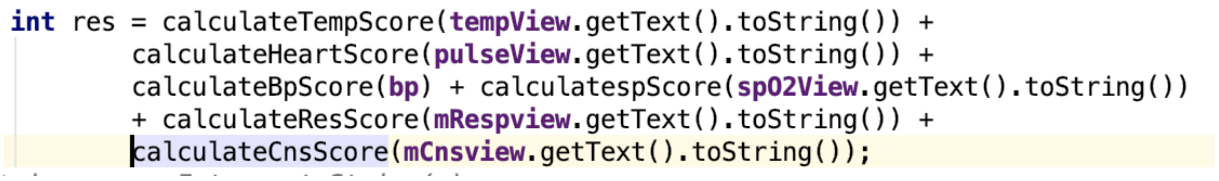
The NEWS score is obtained by the calculation of the sum of each score from the individual vital parameter in accordance with the above table. Below figure 3.20 show the how the score is find out for one parameter which is blood pressure. Here condition are used to decide the score value and same logic can be applied to find score for other parameter too.



```java
private int calculateBpScore(String bp) {
    double bloodPressure = Integer.parseInt(bp);
    int sum = 0;
    if (bloodPressure <= 90) {
        sum += 3;
    } else if ((bloodPressure >= 90.1) && (bloodPressure <= 100)) {
        sum = +2;
    } else if ((bloodPressure >= 100.1) && (bloodPressure <= 110)) {
        sum = +1;
    } else if ((bloodPressure >= 110.1) && (bloodPressure <= 219)) {
        sum = +0;
    } else if ((bloodPressure >= 219.1) && (bloodPressure <= 400)) {
        sum += 3;
    }
    return sum;
}
```

Figure 3.20: Blood Pressure Score Calculation

For example, if a vital reading of the patient is 13 for respiratory rate he got score zero from it, 95 for spo2 than score one, 40 for temperature than score two, 110 for BP than score one, 79 for heart rate than score zero and if patient is alert for APVU than score zero, so the total NEWS score of the patient will be three. Below figure 3.21 show way to sum and get single NEWS score.

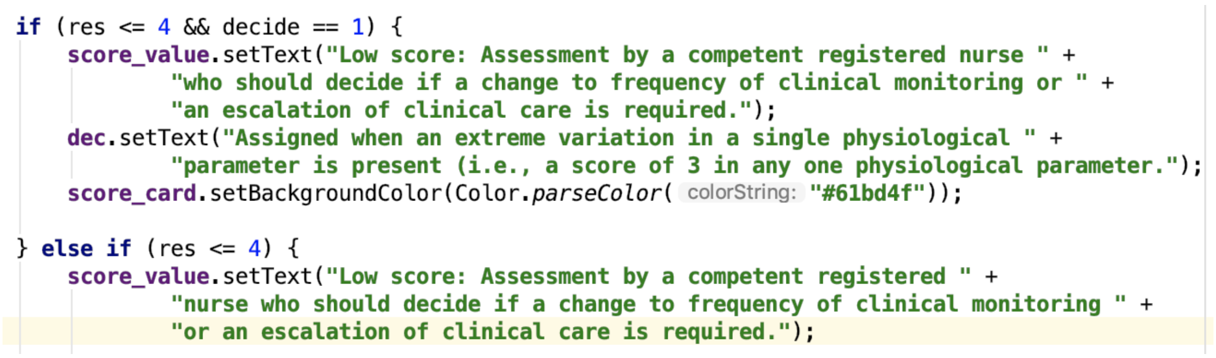


```java
int res = calculateTempScore(tempView.getText().toString()) +
        calculateHeartScore(pulseView.getText().toString()) +
        calculateBpScore(bp) + calculatespScore(sp02View.getText().toString())
        + calculateResScore(mRespview.getText().toString()) +
        calculateCnsScore(mCnsview.getText().toString());
```

Figure 3.21: Summing individual score

Then this score is validated with the score group, and respective care should be provided to the patient according to the scored obtained. The score group is divided into three score group low score, medium score, high score.

- Low Score (NEWS 1-4): When any patient got a low score value than they should be assessed by a health worker or competent registered nurse. Also, they should decide if the change in any parameter of the patient needs further clinical monitoring or an escalation of clinical care is required. Below figure 3.22 shows to validate low score in application and show information, in similar way we can do for other medium score and high too.



```
if (res <= 4 && decide == 1) {
    score_value.setText("Low score: Assessment by a competent registered nurse " +
            "who should decide if a change to frequency of clinical monitoring or " +
            "an escalation of clinical care is required.");
    dec.setText("Assigned when an extreme variation in a single physiological " +
            "parameter is present (i.e., a score of 3 in any one physiological parameter.");
    score_card.setBackgroundColor(Color.parseColor( colorString: "#61bd4f"));

} else if (res <= 4) {
    score_value.setText("Low score: Assessment by a competent registered " +
            "nurse who should decide if a change to frequency of clinical monitoring " +
            "or an escalation of clinical care is required.");
```

Figure 3.22: Low Score Validation

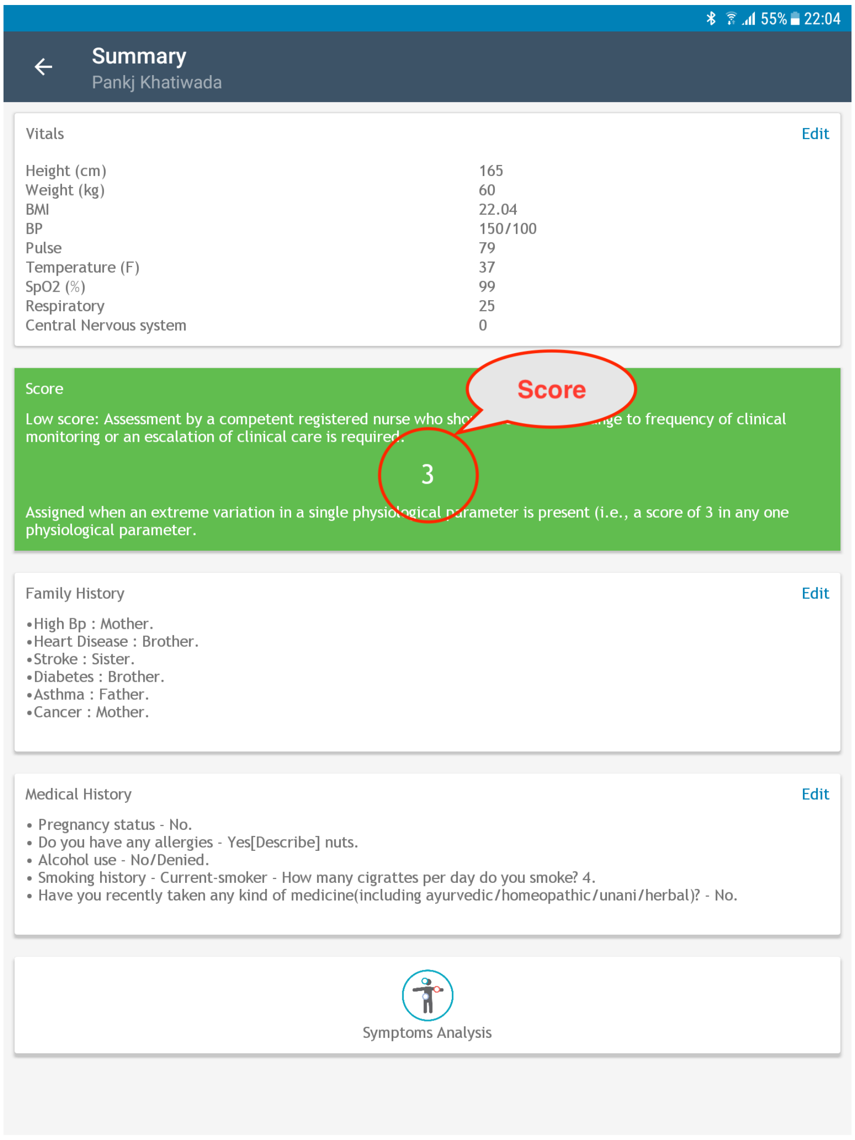
- Medium Score (NEWS 5-6): When any patient got a medium score value than the clinical skilled personal should urgently review him who has the ability in the analysis of acute illness they can be acute team nurse or doctor. We should also review whether the patient needs high-level care and arrange a critical care outreach team.

- High Score (NEWS ≥ 7): When any patient got a high score value than they should immediately be sent for emergency analysis by an expert with an ability of clinical and critical care outreach team, usually patient also need to be transferred to the high level of care area

This shows that the patient with a low score can continue getting normal or as usual care and information but patient with medium and high score should watch more attentively an also should plan for to transfer them on higher care medical unit such as ICU on hospitals.

Apart from these score group, there is also a RED score which refers there is an extreme variation in any one of the parameters such as the value of 3 from the table above on anyone vital parameter. For example, if we have a heart rate of more than 131 or less than 40 than we got the value score three which shows extreme variation in heart rate parameter and is considered the whole as a red score.

In our application, we developed the program to calculate the score based on NEWS table above and displayed the sum score in the patient summary page. When the vitals are taken from the patient, then they are saved in the local database and score is calculated and observation is displayed as shown in figure 3.23.



Summary
Pankj Khatiwada

Vitals Edit

Height (cm)	165
Weight (kg)	60
BMI	22.04
BP	150/100
Pulse	79
Temperature (F)	37
SpO2 (%)	99
Respiratory	25
Central Nervous system	0

Score

Low score: Assessment by a competent registered nurse who sho nge to frequency of clinical monitoring or an escalation of clinical care is required.

3

Assigned when an extreme variation in a single physiological parameter is present (i.e., a score of 3 in any one physiological parameter.

Family History Edit

•High Bp : Mother.
•Heart Disease : Brother.
•Stroke : Sister.
•Diabetes : Brother.
•Asthma : Father.
•Cancer : Mother.

Medical History Edit

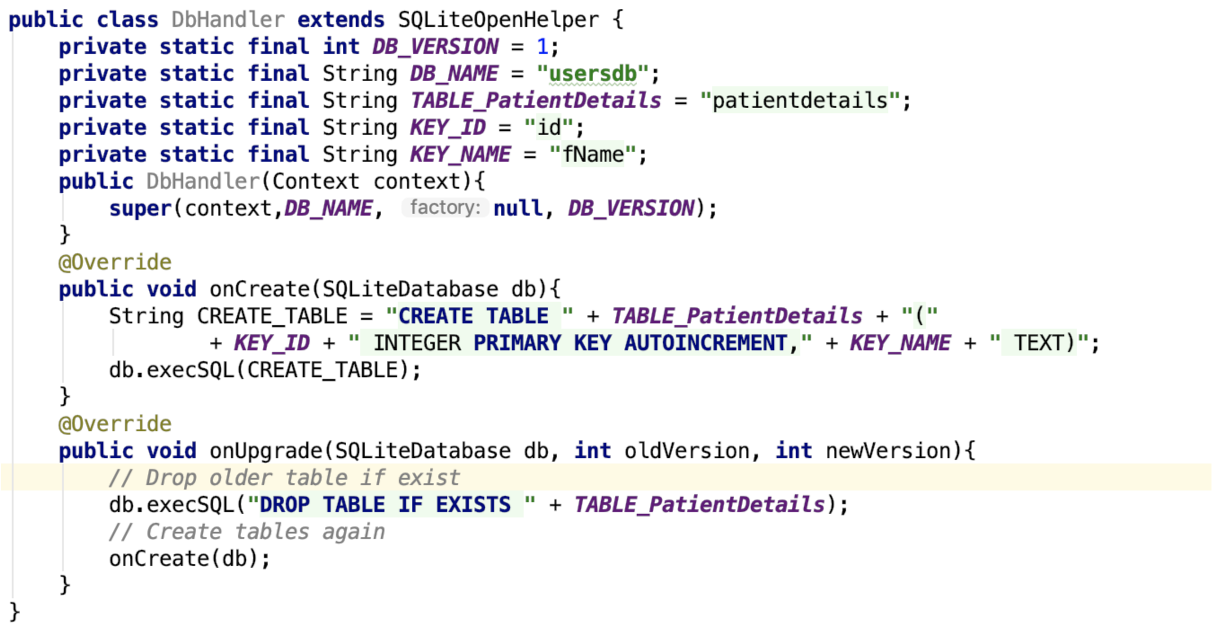
• Pregnancy status - No.
• Do you have any allergies - Yes[Describe] nuts.
• Alcohol use - No/Denied.
• Smoking history - Current-smoker - How many cigrattes per day do you smoke? 4.
• Have you recently taken any kind of medicine(including ayurvedic/homeopathic/unani/herbal)? - No.

Symptoms Analysis

Figure 3.23: Calculated Score

3.2.5 Saving Data in Application

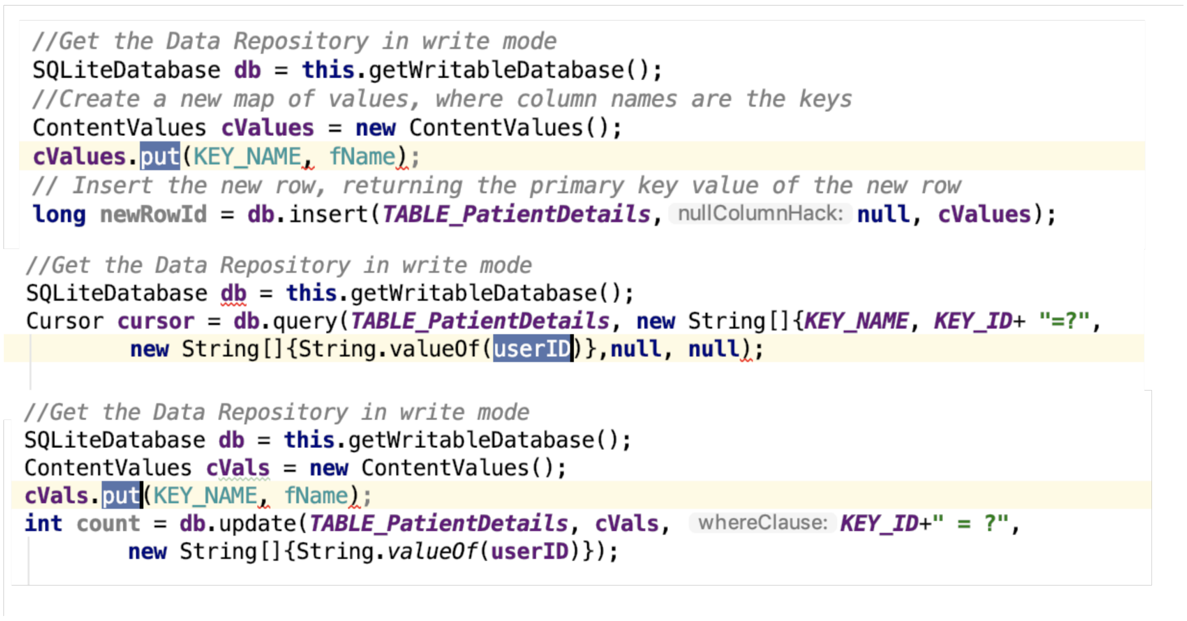
As we know, Android uses built-in SQLite database, in which application-specific data files are stored and inaccessible to other applications. To save the data in the app first we need to create a database and create tables inside it which can store the data. In our app also we need to save a lot of information such as patient details, user details, vital details, score details, observation details, history details etc. So, these can be done by creating tables on the database and saving all the information required for us so that we can retrieve later for our use also. Here we discuss saving the patient name inside the patient details table on our database which gives an overview of how we can save other data in our app in a similar way. So in android by using SQLiteOpenHelper class, we can easily create a database and table for our application. The code snippet on figure 3.24 helps on creating database and tables using class in our Android app. It creates a userdb database and patient details as a table and saves the name of the patient on it.



```
public class DbHandler extends SQLiteOpenHelper {
    private static final int DB_VERSION = 1;
    private static final String DB_NAME = "usersdb";
    private static final String TABLE_PatientDetails = "patientdetails";
    private static final String KEY_ID = "id";
    private static final String KEY_NAME = "fName";
    public DbHandler(Context context){
        super(context,DB_NAME,  factory: null, DB_VERSION);
    }
    @Override
    public void onCreate(SQLiteDatabase db){
        String CREATE_TABLE = "CREATE TABLE " + TABLE_PatientDetails + "("
                + KEY_ID + " INTEGER PRIMARY KEY AUTOINCREMENT," + KEY_NAME + " TEXT)";
        db.execSQL(CREATE_TABLE);
    }
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion){
        // Drop older table if exist
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_PatientDetails);
        // Create tables again
        onCreate(db);
    }
}
```

Figure 3.24: Create DB & Table

Now for to save or insert the new patient name in the table we use the insert() method, read patient from the database we use query() method and to update patient name we use update() method as shown in the following figure 3.25.



```
//Get the Data Repository in write mode
SQLiteDatabase db = this.getWritableDatabase();
//Create a new map of values, where column names are the keys
ContentValues cValues = new ContentValues();
cValues.put(KEY_NAME, fName);
// Insert the new row, returning the primary key value of the new row
long newRowId = db.insert(TABLE_PatientDetails, nullColumnHack: null, cValues);
```

```
//Get the Data Repository in write mode
SQLiteDatabase db = this.getWritableDatabase();
Cursor cursor = db.query(TABLE_PatientDetails, new String[]{KEY_NAME, KEY_ID+ "=?",
        new String[]{String.valueOf(userID)},null, null);
```

```
//Get the Data Repository in write mode
SQLiteDatabase db = this.getWritableDatabase();
ContentValues cVals = new ContentValues();
cVals.put(KEY_NAME, fName);
int count = db.update(TABLE_PatientDetails, cVals, whereClause: KEY_ID+" = ?",
        new String[]{String.valueOf(userID)});
```

Figure 3.25: Insert, Read and Update Detail

.

Chapter 4

Result

In this chapter we will be discussing about the results of our application mTeleHealth-UiA obtained through our design and solution from above. Here we will explain the different activity and the interface of app such as form to receive medical data from PHD, to save patient personal and medical information , displaying NEWS score, symptoms assessment and providing suggestion accordingly.

4.1 Login and Registration

When the user first opens the app, the splash screen appears as shown in figure 4.1 screen 1. After that the login page is shown as 4.1 screen 2, if the health worker is already registered, then they can login by entering their credentials email and password which will be then checked with the previous register profile saved in the SQLite database in the Profile table. If not they can register by clicking on the link *No account yet? Create One* which then opens the registration page as shown in figure 4.1 screen 3 then the profile is saved in the SQLite database in the table.

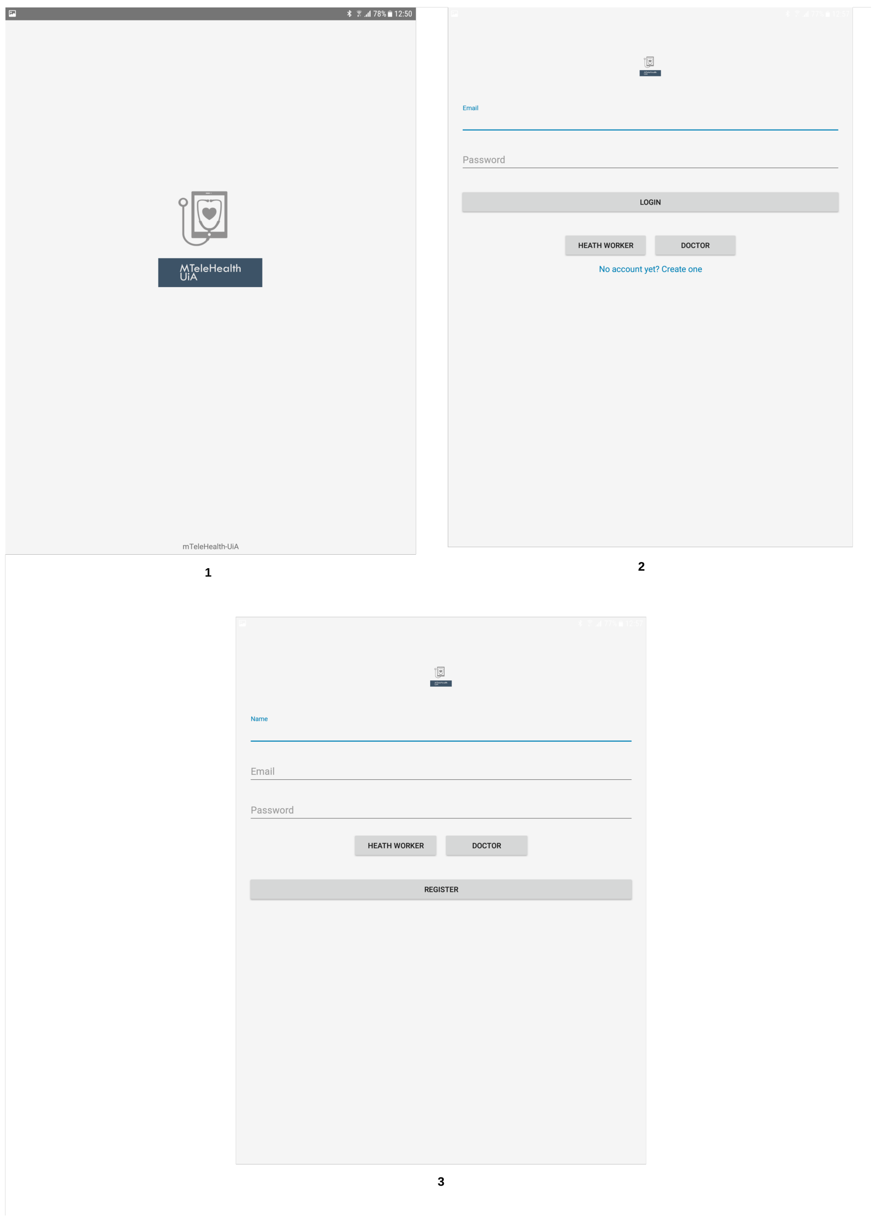
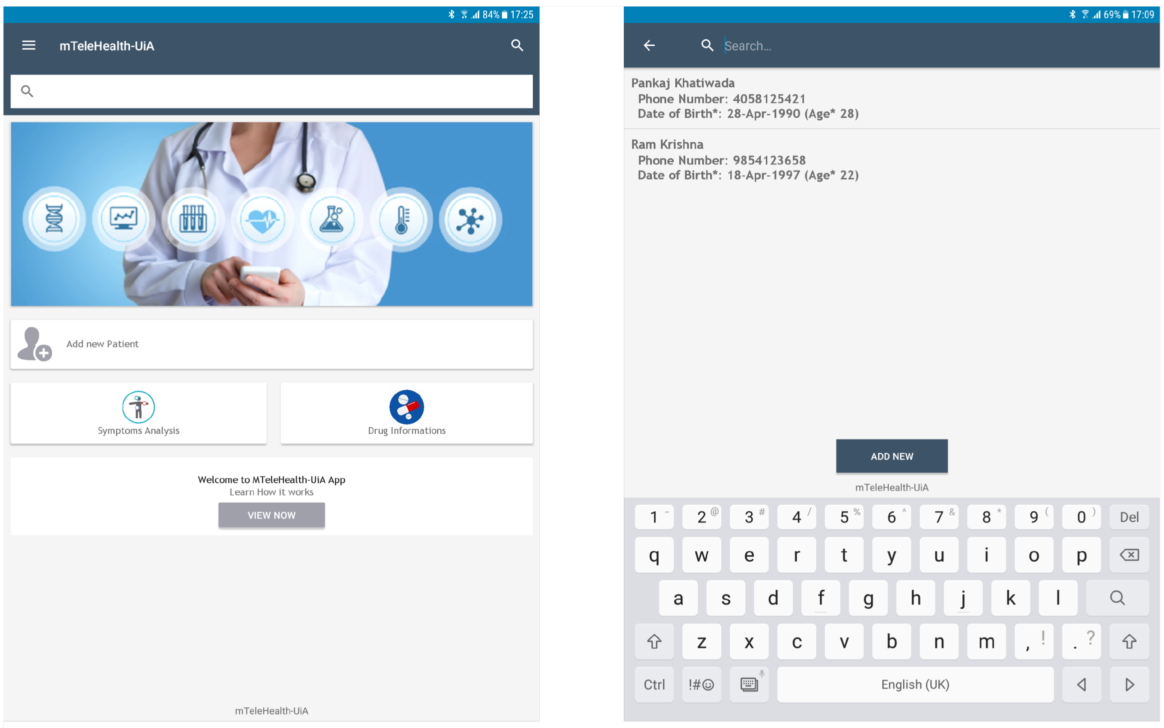


Figure 4.1: Login and Signup

4.2 Homepage

When the login is successful than the respective homepage is opened according to role, we can have two option for homepage either of health worker or a doctor according to the saved data during registration. Here we deal with only health worker, so health worker home appears as shown in figure 4.2a. The doctor home page is for the extension of the project, so only the wireframe of the page is drawn and planned for the future.

As we can see, there is an Add new Patient button, which we can select to register the new patient. If the patient is already registered, then we can search him from the search field



Pankaj Khatiwada
Phone Number: 4058125421
Date of Birth*: 28-Apr-1990 (Age* 28)

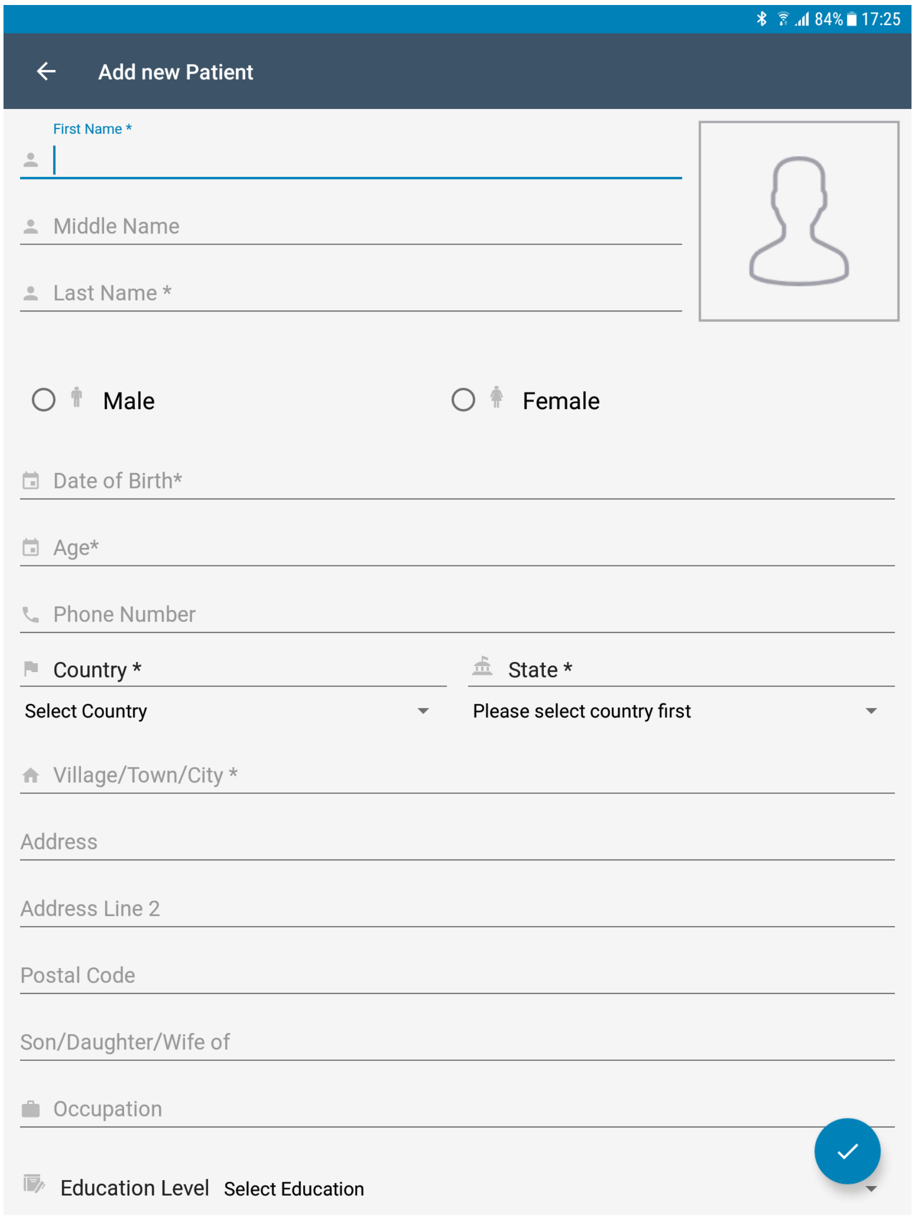
Ram Krishna
Phone Number: 9854123658
Date of Birth*: 18-Apr-1997 (Age* 22)

Figure 4.2: Homepage and Search

by typing his name as shown in figure 4.2b. We also have a symptom assessment button, which gives the information of the patient medical condition by examining the symptoms. Also, we got drug information button which is later extension of the project where we provide information about different drugs. Also, we have a section where we have button VIEW NOW contains the information about how the app works for the first-time app user.

4.3 Registering Patient Information

The individual patient information is recorded by a health worker using mTeleHealth app, which contains patient personal information. It is a similar process as registering the patient in the traditional paper files but with app it is recorded electronically and saving the information on the database where it can be retrieved later for further use. In our application we store the patient information using the screen interface as shown in below figure 4.3


84% 17:25

Add new Patient

First Name *

Middle Name

Last Name *

Male

Female

Date of Birth*

Age*

Phone Number

Country *

Select Country

State *

Please select country first

Village/Town/City *

Address

Address Line 2

Postal Code

Son/Daughter/Wife of

Occupation

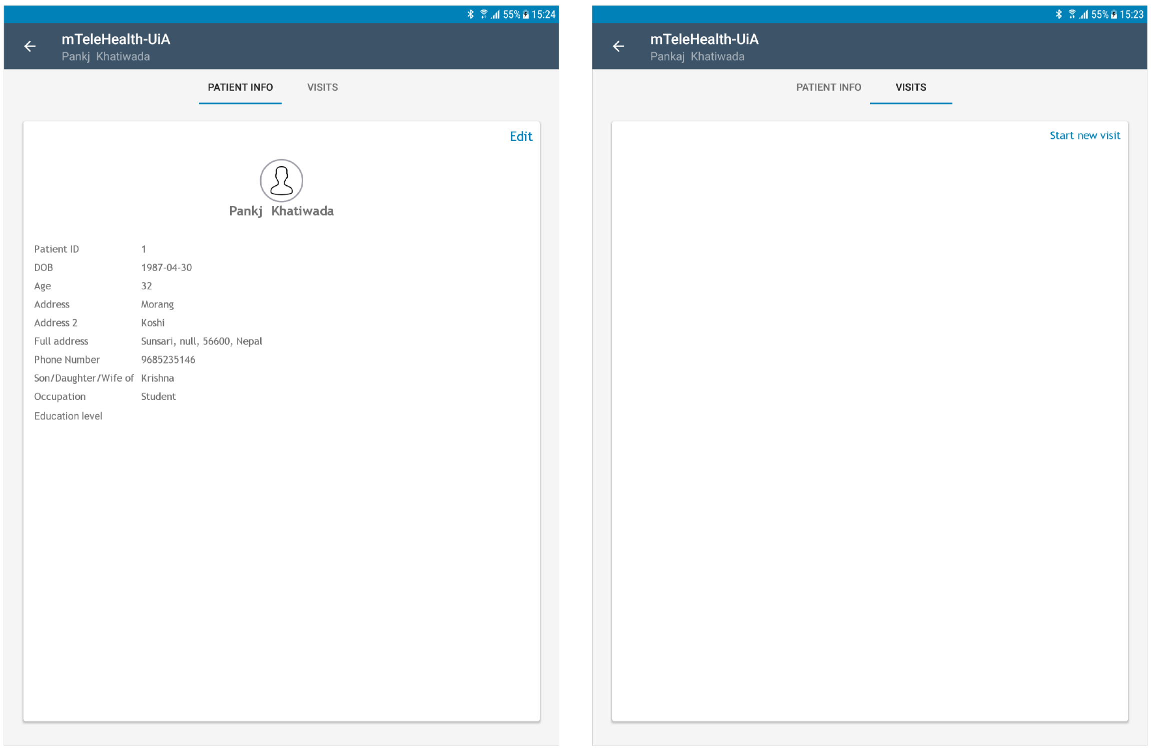
Education Level Select Education

Figure 4.3: Patient Record Page

When we click add patient on the home screen,below screen is opened and health worker filled out all the necessary information as shown in screen and is saved by clicking the blue circle button which is on the right side on the bottom of the page. When the button is clicked the data are saved into database.

4.4 Patient Summary

After saving the details of the patient in the application, we will reach on patient summary activity where we can see two tabs patient info and visit as shown in figure 4.4a. If we want to edit the patient information, we can do it by pressing the Edit in the screen which opens the Patient Record Page and we can edit as per requirement.



Patient ID	1
DOB	1987-04-30
Age	32
Address	Morang
Address 2	Koshi
Full address	Sunsari, null, 56600, Nepal
Phone Number	9685235146
Son/Daughter/Wife of	Krishna
Occupation	Student
Education level	

a

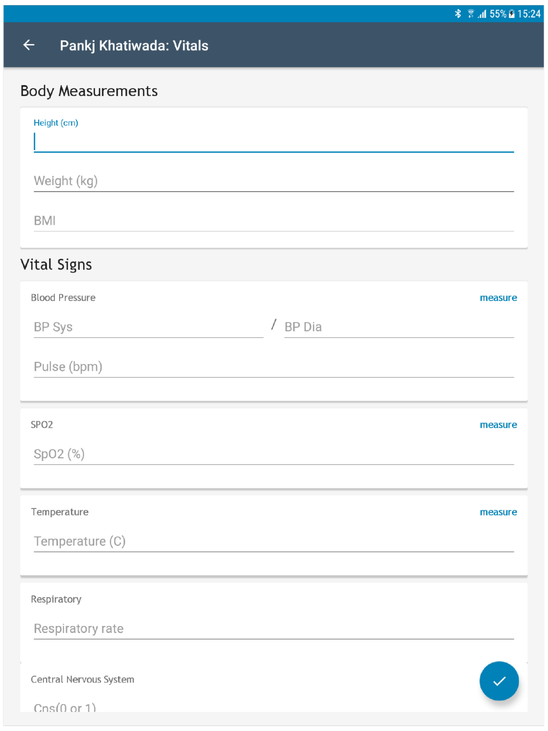
b

Figure 4.4: Patient Summary and Visit Page

The next tab is visit tab which is shown in figure 4.4b which now is empty, there will be a patient visit record after the patient visit is completed. To start a patient visit we click on the Start New Visit on the activity.

4.5 Recording Vitals

When the new visit of patient is started first the vitals of the patients are recorded. The vitals record form is shown in the figure 4.5a. The vitals of the patients here can be recorded manually also automatically from medical devices. The reading from the medical devices are transferred automatically via BLE or Bluetooth on the application as on figure 4.5b and are saved for the calculation of the NEWS score which will be shown on the patient visit summary.


55% 15:24

Pankj Khatiwada: Vitals

Body Measurements

Height (cm)

Weight (kg)

BMI

Vital Signs

Blood Pressure measure

BP Sys / BP Dia

Pulse (bpm)

SPO2 measure

SpO2 (%)

Temperature measure

Temperature (C)

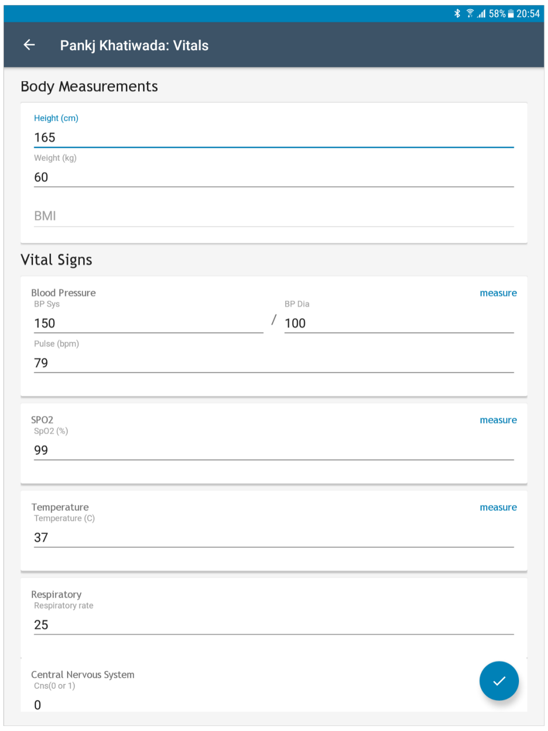
Respiratory

Respiratory rate

Central Nervous System

Cns(0 or 1)

a


58% 20:54

Pankj Khatiwada: Vitals

Body Measurements

Height (cm): 165

Weight (kg): 60

BMI

Vital Signs

Blood Pressure measure

BP Sys: 150 / BP Dia: 100

Pulse (bpm): 79

SPO2 measure

SpO2 (%): 99

Temperature measure

Temperature (C): 37

Respiratory

Respiratory rate: 25

Central Nervous System

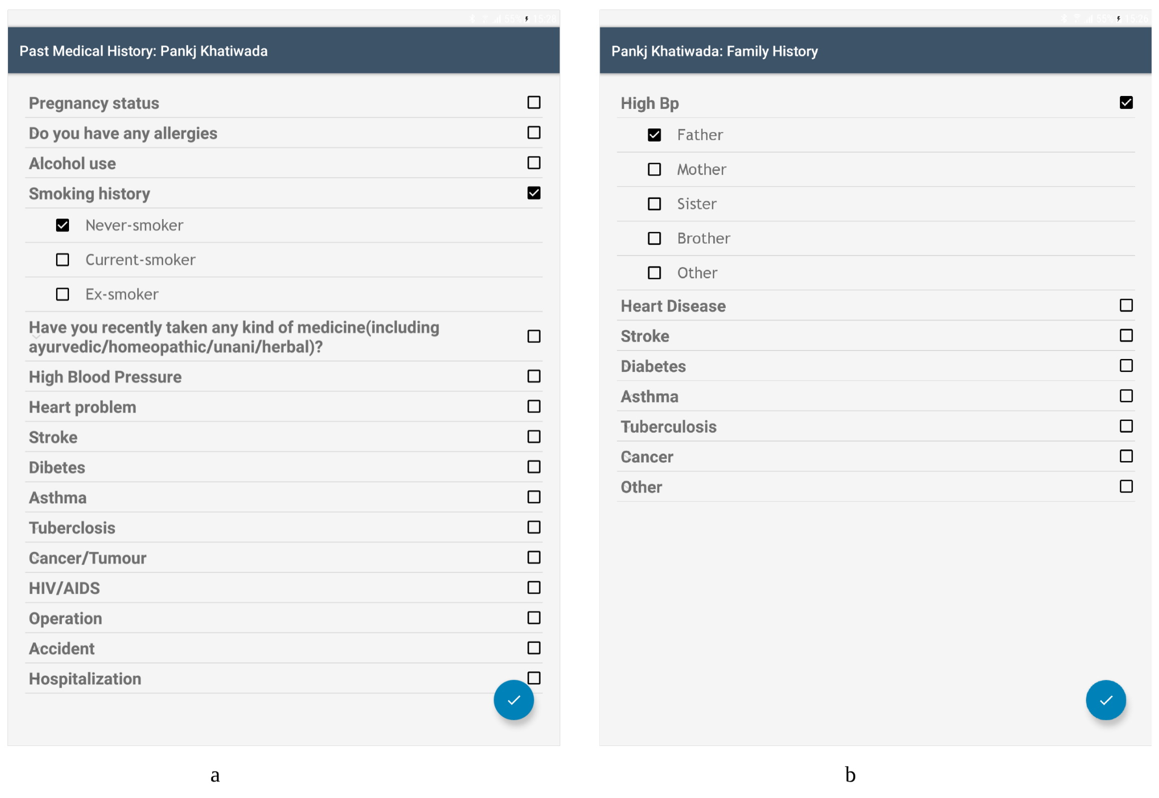
Cns(0 or 1): 0

b

Figure 4.5: Record Vitals Page

4.6 Family and Medical History of Patient

After the vitals are taken the new activity is opened as shown in the figure 4.6 which then takes the past medical history of the patient as in 4.6a and family history of the patient as in 4.6b. All the selected medical and family history of the patient are saved in the database and they are displayed in patient visit summary.



Past Medical History: Pankj Khatiwada

Pregnancy status	☐
Do you have any allergies	☐
Alcohol use	☐
Smoking history	☑
☑ Never-smoker	
☐ Current-smoker	
☐ Ex-smoker	
Have you recently taken any kind of medicine(including ayurvedic/homeopathic/unani/herbal)?	☐
High Blood Pressure	☐
Heart problem	☐
Stroke	☐
Dibetes	☐
Asthma	☐
Tuberclosis	☐
Cancer/Tumour	☐
HIV/AIDS	☐
Operation	☐
Accident	☐
Hospitalization	☐

a

Pankj Khatiwada: Family History

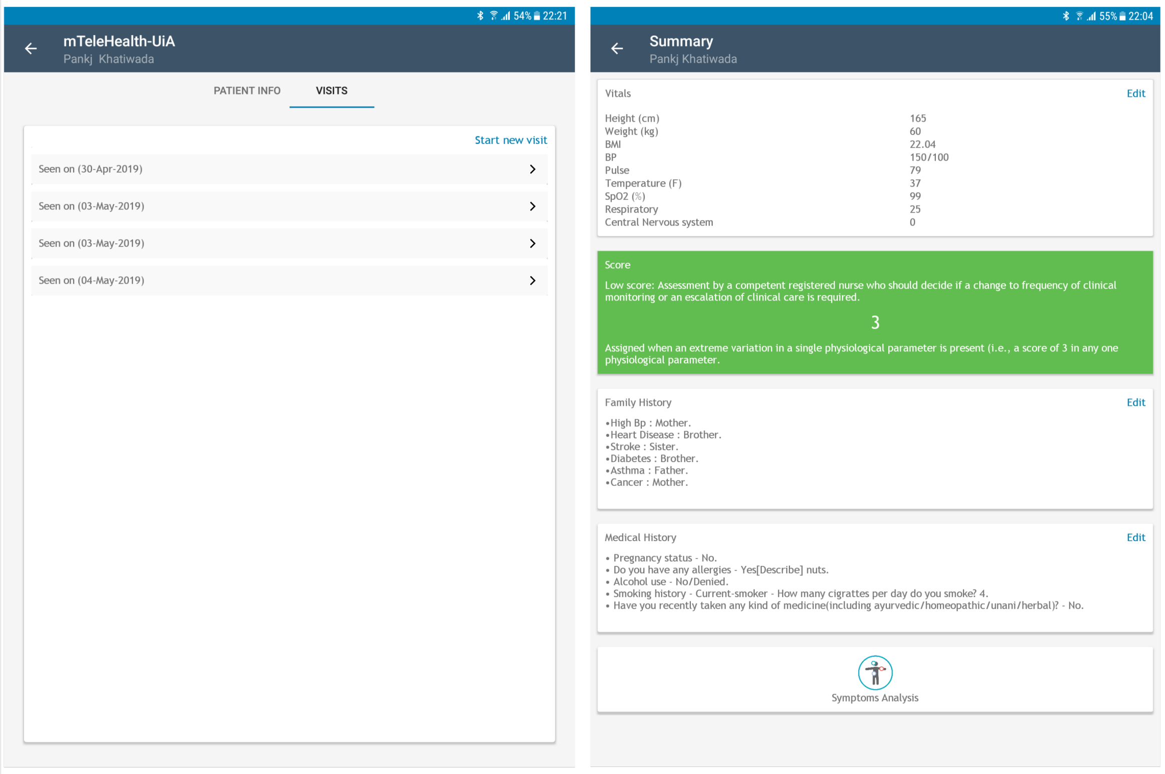
High Bp	☑
☑ Father	
☐ Mother	
☐ Sister	
☐ Brother	
☐ Other	
Heart Disease	☐
Stroke	☐
Diabetes	☐
Asthma	☐
Tuberculosis	☐
Cancer	☐
Other	☐

b

Figure 4.6: Family and Medical History

4.7 Patient Visit Summary

When all the information of the patient are recorded such as a vital sign, family and medical history all data are saved and later displayed on the patient summary as in figure 4.6b in the figure 4.6a we see the dates which are the patient visit history dates. Each of the dates includes independent patient visit summary. The patient summary in figure 4.6b include vital sign summary on the top. After that, it contains the score calculated as the basis of NEWS score chart and gives some recommendation. After that, we can see the family and medical history of the patient. We can also edit the respective value by pressing on Edit on each section. Also, at last, we can see the symptom assessment button, which is used to get the overview of the patient health status and some recommendation by the application.


54% 22:21

mTeleHealth-UiA
Pankj Khatiwada

PATIENT INFO | VISITS

Start new visit

Seen on (30-Apr-2019)

Seen on (03-May-2019)

Seen on (03-May-2019)

Seen on (04-May-2019)

a

55% 22:04

Summary
Pankj Khatiwada

Vitals Edit

Height (cm)	165
Weight (kg)	60
BMI	22.04
BP	150/100
Pulse	79
Temperature (F)	37
SpO2 (%)	99
Respiratory	25
Central Nervous system	0

Score

Low score: Assessment by a competent registered nurse who should decide if a change to frequency of clinical monitoring or an escalation of clinical care is required.

3

Assigned when an extreme variation in a single physiological parameter is present (i.e., a score of 3 in any one physiological parameter.

Family History Edit

•High Bp : Mother.
•Heart Disease : Brother.
•Stroke : Sister.
•Diabetes : Brother.
•Asthma : Father.
•Cancer : Mother.

Medical History Edit

- Pregnancy status - No.
- Do you have any allergies - Yes[Describe] nuts.
- Alcohol use - No/Denied.
- Smoking history - Current-smoker - How many cigrattes per day do you smoke? 4.
- Have you recently taken any kind of medicine(including ayurvedic/homeopathic/unani/herbal)? - No.

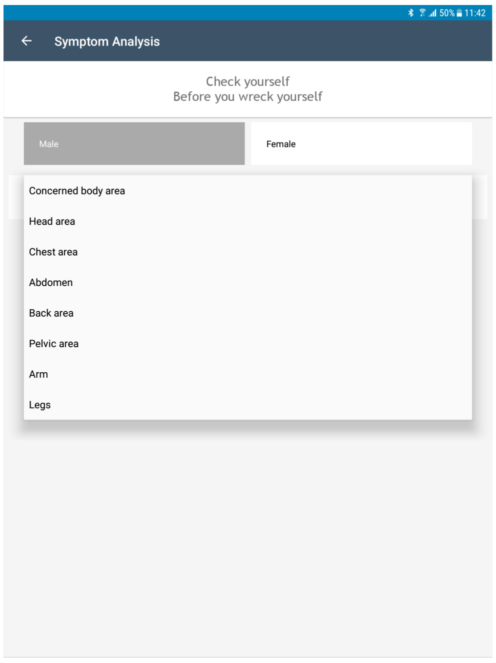
Symptoms Analysis

b

Figure 4.7: Patient Visit Summary

4.8 Patient Symptom Assessment

When the Symptom Analysis button is clicked on the patient visit summary page we got the option as shown in figure 4.8 for the selection of gender and which specific parts of the body is not feeling well. As in figure, we see the male patient is not feeling well in his head area in the nose part. After selection of the nose as in figure 4.8b, we will be given a list of symptoms. We can see a list of symptoms in 4.9 related to the nose and Runny Nose symptom is selected in 4.9a which then shows more specific symptoms related on 4.9b. Now among the various specific symptom, the common cold is clicked, which is similar to the patient symptom. Then we will get the overview information of the disease their related symptoms, prevention measure, how it is treated, how can it be worsen should the patient needs to visit doctor or not,how common the disease is, how to do self care and many more information as shown in figure 4.10. This information helps the health worker to give suggestion and feedback to the patient and can do onsite consultation.



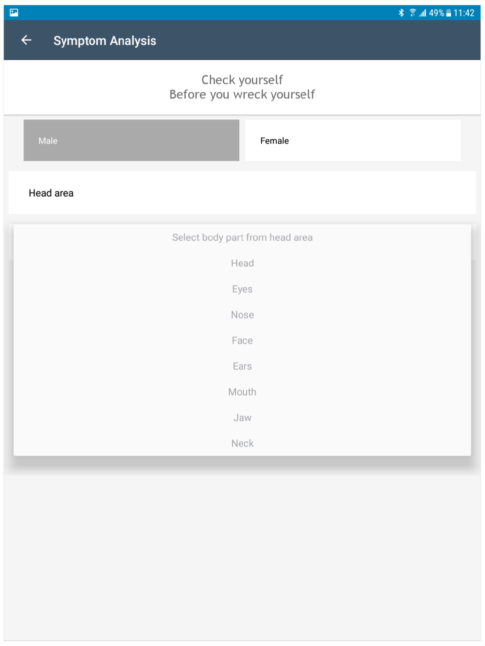
Symptom Analysis

Check yourself
Before you wreck yourself

Male Female

Concerned body area
Head area
Chest area
Abdomen
Back area
Pelvic area
Arm
Legs

a



Symptom Analysis

Check yourself
Before you wreck yourself

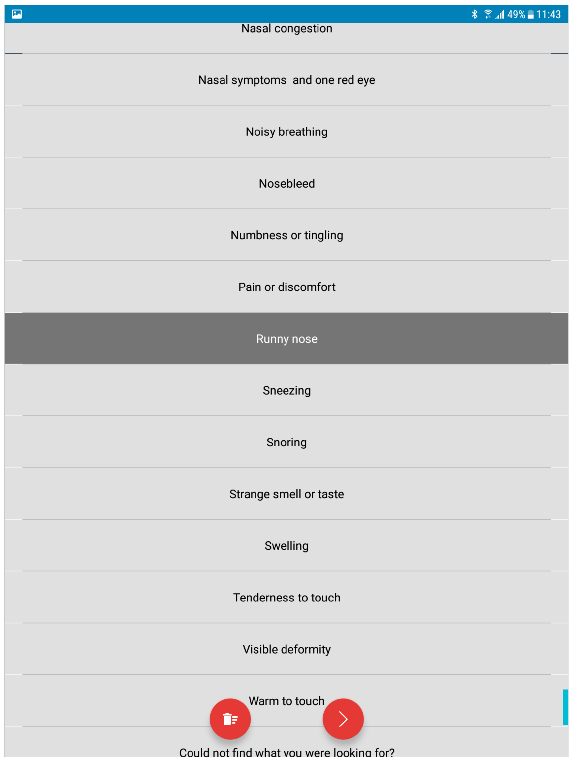
Male Female

Head area

Select body part from head area
Head
Eyes
Nose
Face
Ears
Mouth
Jaw
Neck

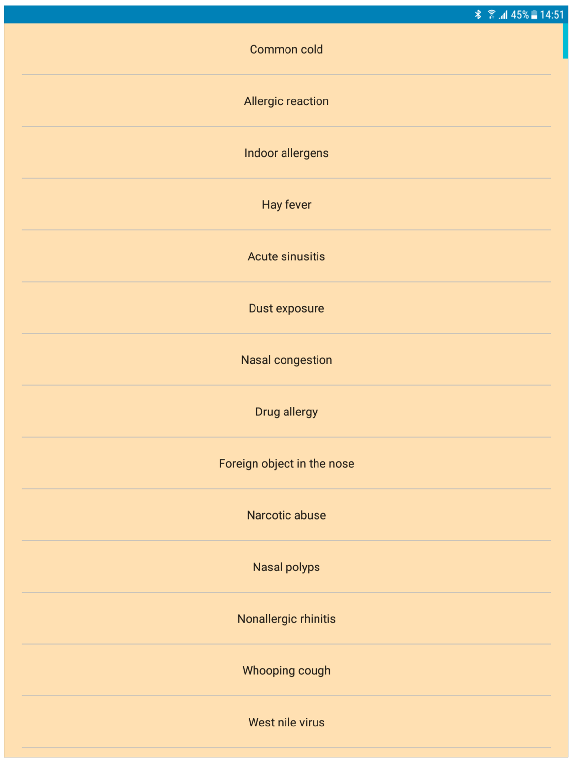
b

Figure 4.8: Body Part Selection



Nasal congestion
Nasal symptoms and one red eye
Noisy breathing
Nosebleed
Numbness or tingling
Pain or discomfort
Runny nose
Sneezing
Snoring
Strange smell or taste
Swelling
Tenderness to touch
Visible deformity
Warm to touch
Could not find what you were looking for?

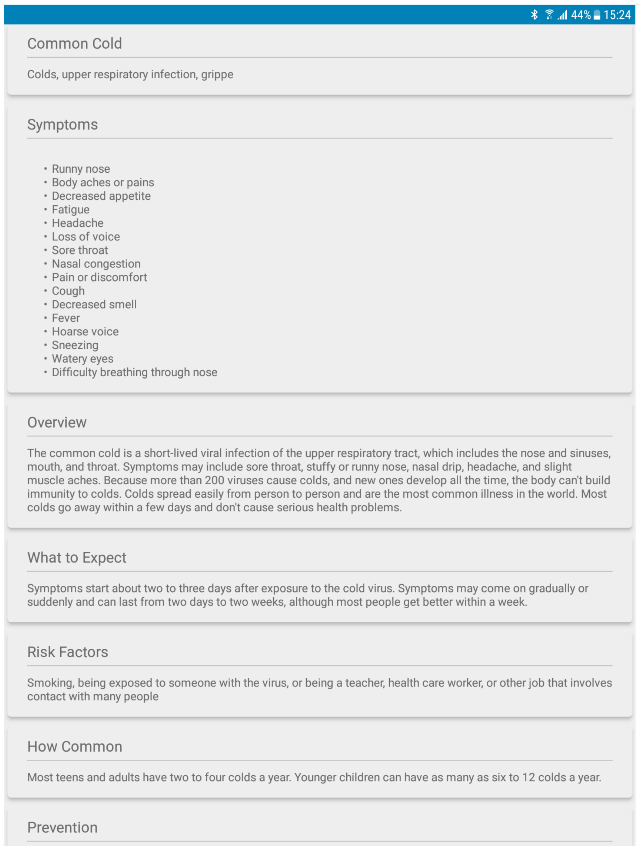
a



Common cold
Allergic reaction
Indoor allergens
Hay fever
Acute sinusitis
Dust exposure
Nasal congestion
Drug allergy
Foreign object in the nose
Narcotic abuse
Nasal polyps
Nonallergic rhinitis
Whooping cough
West nile virus

b

Figure 4.9: Specific Symptom



Common Cold

Colds, upper respiratory infection, grippe

Symptoms

- Runny nose
- Body aches or pains
- Decreased appetite
- Fatigue
- Headache
- Loss of voice
- Sore throat
- Nasal congestion
- Pain or discomfort
- Cough
- Decreased smell
- Fever
- Hoarse voice
- Sneezing
- Watery eyes
- Difficulty breathing through nose

Overview

The common cold is a short-lived viral infection of the upper respiratory tract, which includes the nose and sinuses, mouth, and throat. Symptoms may include sore throat, stuffy or runny nose, nasal drip, headache, and slight muscle aches. Because more than 200 viruses cause colds, and new ones develop all the time, the body can't build immunity to colds. Colds spread easily from person to person and are the most common illness in the world. Most colds go away within a few days and don't cause serious health problems.

What to Expect

Symptoms start about two to three days after exposure to the cold virus. Symptoms may come on gradually or suddenly and can last from two days to two weeks, although most people get better within a week.

Risk Factors

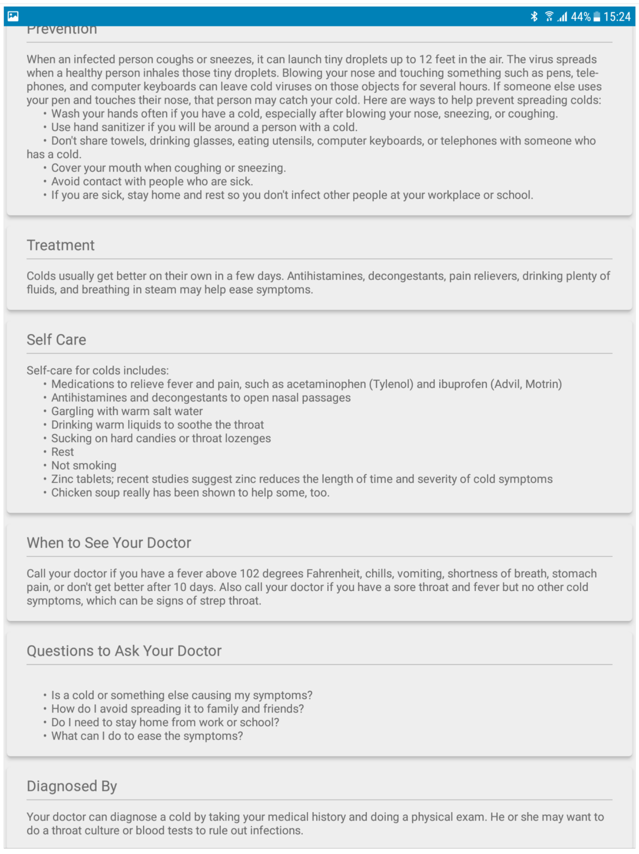
Smoking, being exposed to someone with the virus, or being a teacher, health care worker, or other job that involves contact with many people

How Common

Most teens and adults have two to four colds a year. Younger children can have as many as six to 12 colds a year.

Prevention

a



Prevention

When an infected person coughs or sneezes, it can launch tiny droplets up to 12 feet in the air. The virus spreads when a healthy person inhales those tiny droplets. Blowing your nose and touching something such as pens, telephones, and computer keyboards can leave cold viruses on those objects for several hours. If someone else uses your pen and touches their nose, that person may catch your cold. Here are ways to help prevent spreading colds:

- Wash your hands often if you have a cold, especially after blowing your nose, sneezing, or coughing.
- Use hand sanitizer if you will be around a person with a cold.
- Don't share towels, drinking glasses, eating utensils, computer keyboards, or telephones with someone who has a cold.
- Cover your mouth when coughing or sneezing.
- Avoid contact with people who are sick.
- If you are sick, stay home and rest so you don't infect other people at your workplace or school.

Treatment

Colds usually get better on their own in a few days. Antihistamines, decongestants, pain relievers, drinking plenty of fluids, and breathing in steam may help ease symptoms.

Self Care

Self-care for colds includes:

- Medications to relieve fever and pain, such as acetaminophen (Tylenol) and ibuprofen (Advil, Motrin)
- Antihistamines and decongestants to open nasal passages
- Gargling with warm salt water
- Drinking warm liquids to soothe the throat
- Sucking on hard candies or throat lozenges
- Rest
- Not smoking
- Zinc tablets; recent studies suggest zinc reduces the length of time and severity of cold symptoms
- Chicken soup really has been shown to help some, too.

When to See Your Doctor

Call your doctor if you have a fever above 102 degrees Fahrenheit, chills, vomiting, shortness of breath, stomach pain, or don't get better after 10 days. Also call your doctor if you have a sore throat and fever but no other cold symptoms, which can be signs of strep throat.

Questions to Ask Your Doctor

- Is a cold or something else causing my symptoms?
- How do I avoid spreading it to family and friends?
- Do I need to stay home from work or school?
- What can I do to ease the symptoms?

Diagnosed By

Your doctor can diagnose a cold by taking your medical history and doing a physical exam. He or she may want to do a throat culture or blood tests to rule out infections.

b

Figure 4.10: Patient Observation Details

.

Chapter 5

Evaluation

5.1 Testing of Application

The testing of the app was initially planned to be done on some rural areas in the country of India with some real Health worker involvement, but due to the time limitation, we are unable to deploy it. So simple test was made our-self with the help of ten different participants, as we made a simple authentication data for application login username as "healthworker" and password as "123456" for the health worker. The clear demonstration of the app was done with a detailed explanation of how the application works. For testing, we collect answers from ten different users whom we provide the app and ask them to install on their phone, record some patient information and their vital sign, take a history of the patient and observe symptom analysis. After that we did some quick survey on them and asked some few questions about apps:

Question1: How easy was it to install our mTeleHealth-UiA apps?
The app needs to install on the devices to use it. So we provide all the participants with the apk which is also an application package file that can be saved on memory and with the help of file browser it can be installed on the device following simple installation procedure as other regular application. The below figure 5.1 shows the result of the survey.

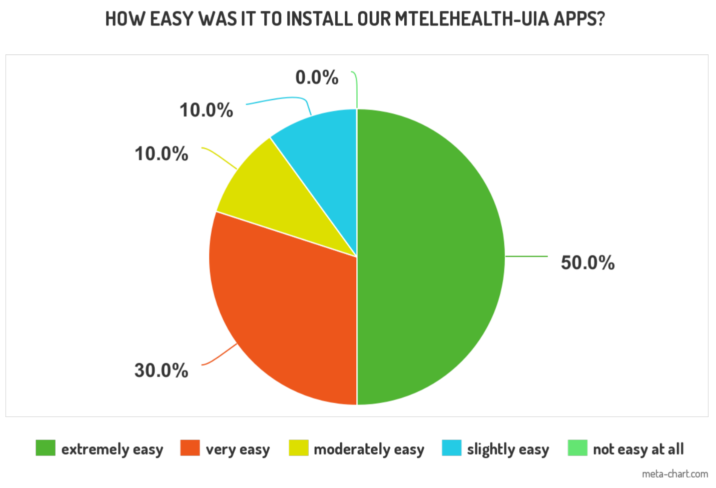


Figure 5.1: Survey from Question 1

Question2: How user-friendly is our application system interface?

This question clarifies that whether the app user interface is handy for it user or not and also to ensure the design requirement that was focused on easy to use, clarity, learn and understand the app while using. It is the developer interest to know whether the users found our app systems interface friendly or not. The below figure 5.2 shows the result of the survey.

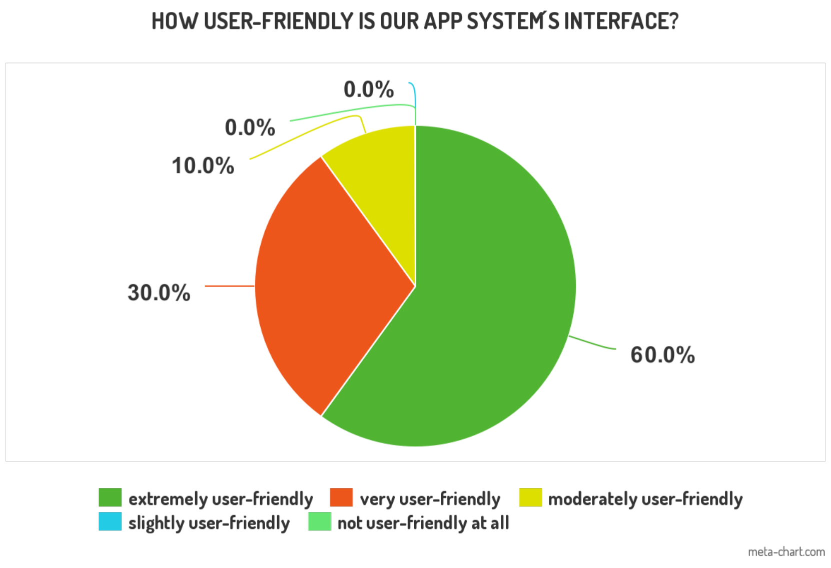


Figure 5.2: Survey from Question 2

Question3: How often does our app freeze or crash while using?

After installation and using the application in devices sometimes they get crash and freeze while using. Identifying the efficiency of the app was necessary so that the bugs on the application will be fixed in near future. The below figure 5.3 shows the result of the survey.

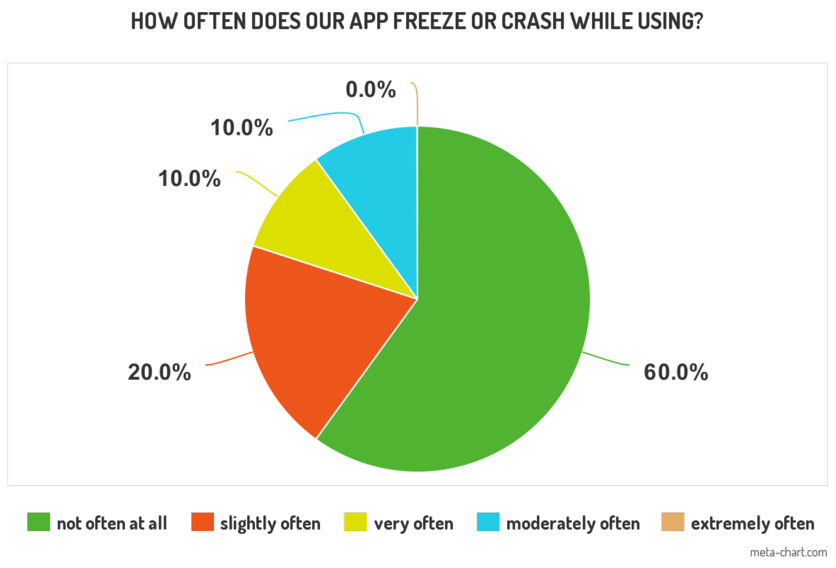


Figure 5.3: Survey from Question 3

Question4: How successful is our app in performing its intended task?
It was necessary to identify whether the smartphone based app was implemented the way it was designed to meet our research problem. For this, another survey question was prepared, whether the application perform its intended task successfully or not. The below figure 5.4 shows the result of the survey.

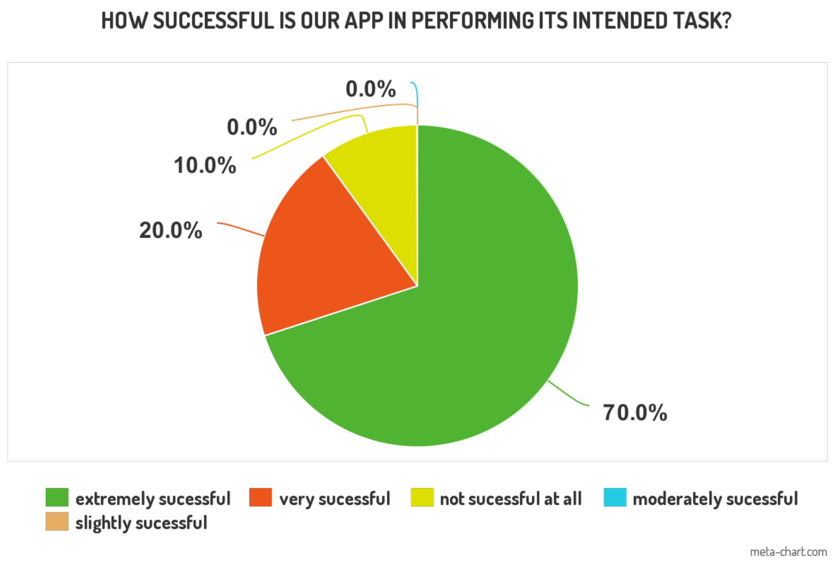


Figure 5.4: Survey from Question 4

Question5: Do you think this app provide ease way on measuring and saving

vital parameter of patient?

Most user were satisfied on the process of obtaining the data from the medical devices to our app. User simply need to click measure button and after measuring the data displayed on medical devices were automatically transferred to the app via Bluetooth. This makes user smooth process on getting vital data. The below figure 5.5 shows the result of the survey.

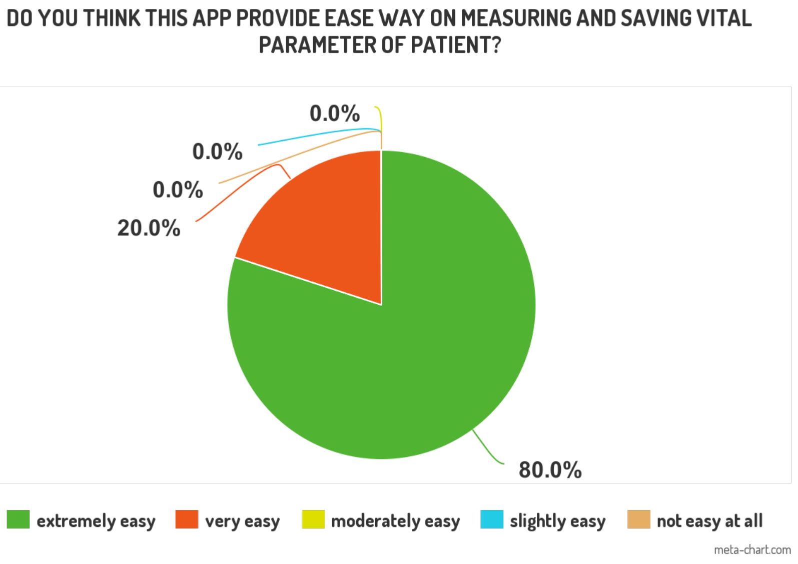


Figure 5.5: Survey from Question 5

Question6: Do you think this app provide necessary information of NEWS and symptom analysis?

The user are able to obtain the score from vital parameter and are satisfied by its recommendation. Also symptom analysis tool they thought to be great to use before referring the patient to the doctor because it gives overview and medical condition if the patient.The below figure 5.6 shows the result of the survey.

Question7: Do you think this app is useful for collecting the patient information?

It is necessary to collect the information from the patient as much as we can because it helps for to decide what further step should be taken and also for smooth diagnosis for the patient. So, we asked user how smooth was the process on collecting all these information from patient such as patient, vital, history and symptoms. The below figure 5.7 shows

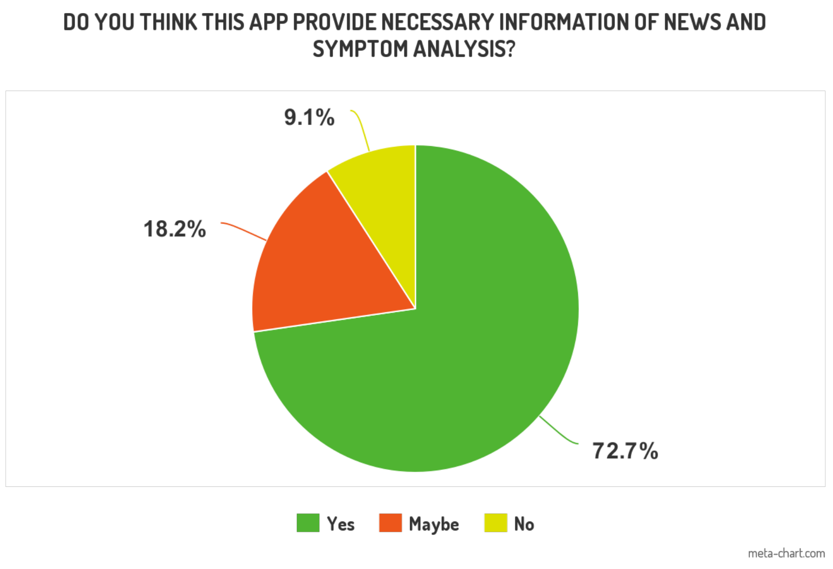


Figure 5.6: Survey from Question 6

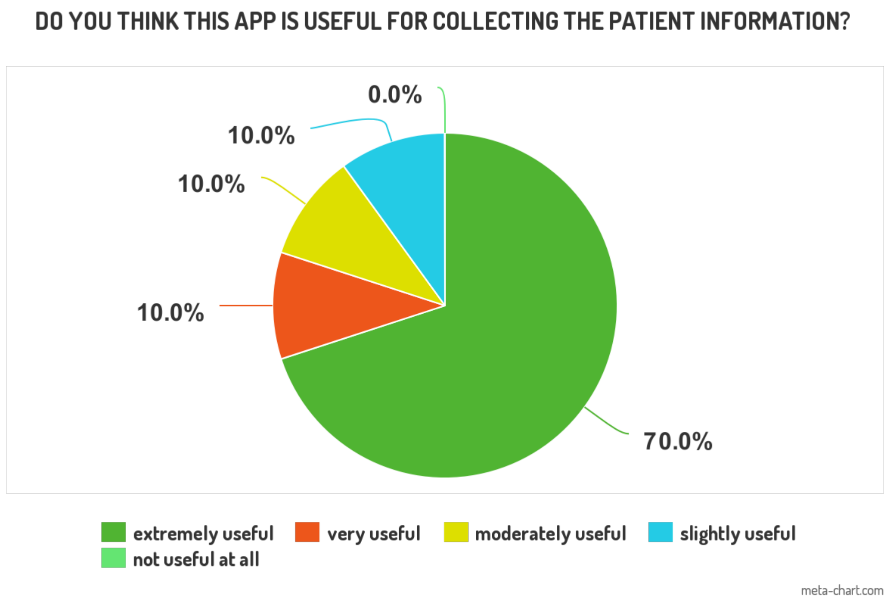


Figure 5.7: Survey from Question 7

the result of the survey.

Question8: Do you think this app is ready for to go trial in rural areas?

Our main aim for the development of this app on this thesis is to go on trial phase with the help of rural area health worker and to see how it perform. So, we ask the users is app somehow ready and good for to go on trial phase to help health worker in rural areas. The below figure 5.8 shows the result of the survey.

.

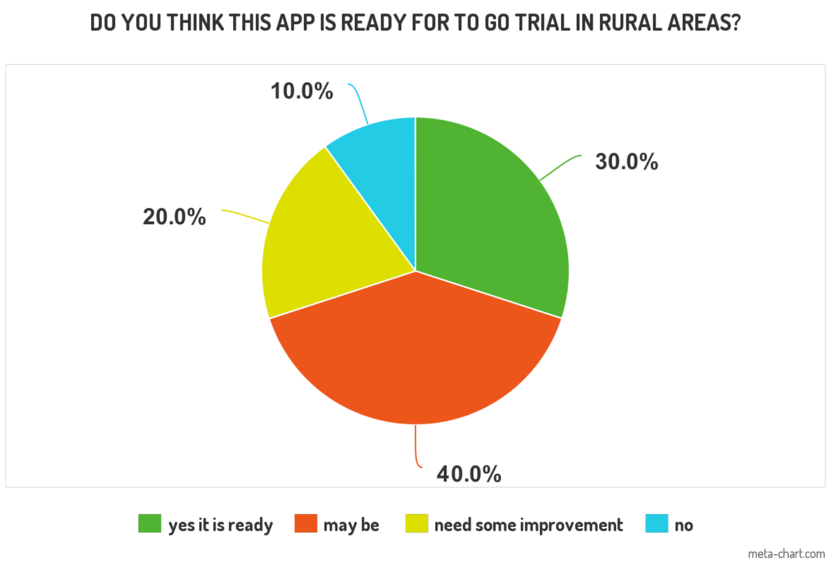


Figure 5.8: Survey from Question 8

Chapter 6

Discussion

This chapter includes the discussion related with the development of the technology within the healthcare system and how did our application helps in advancing the health status of people in rural areas with the help of health worker. We also discuss about our research questions and does this application fulfil the demand what are required.

6.1 Health and Technology

Technology nowadays is rapidly growing in the health sector in the aid that it will make our life easier. It has also played a vital role in the areas where there is no availability of health care facilities such as remote areas with the help of health application for handheld devices. Nowadays we can find apps for almost every health-related types, but also we should be aware of its authenticity and trustworthiness. There are fitness app, stress management app, weight losing app, symptom analysis app, medication app, an app for measuring different body parameter, disease monitoring app, telehealth apps, and many others. People nowadays can monitor their vital parameters by themselves by medical sensor devices (PHD) which can be readily available in the market. People can also get information about their disease and symptoms with the help of eHealth application that is available in the market and even on the internet. Nowadays if people got sick, they often search on mHealth apps or online about their condition before visiting the doctors. But what if the PHD available in the market are not certified and provide unreliable reading what if the information available in the application or online are not trustworthy which leads to high risk of misdiagnosis. When it comes to the diagnosis or to give a recommendation, the information provided by apps should be accurate and reliable and be from an authentic source. So, here we used the Continua certified products for vitals measurement and for score calculation we adopt the NHS (UK) chart and for building symptoms knowledge base we used a source from WebMD, MayoClinic, Healthline etc which are trusted by many medical doctors. Although there are many mHealth applications they are usually focused on a single patient there is not much application for health worker by which they can collect information from the patient and can give some preliminary instruction as advice by the application. The collected data can also be essential aspects for the further diagnosis of the patient by doctors and get an overview to decide what should be done next for the patient. So we had discussed all these matters and develop the app focused on the rural areas health worker which can be able to carry out these work on this thesis work.

6.2 Research Question Discussion

Now here we discuss about our research questions from the introduction section and what steps we had taken to achieve these task during our thesis.

1. How the application support the health care service which provides decision support and recommendation with the help of technology?

   Here to develop the framework or application we choose the Android platform where we can develop the apps for Android smartphones or tablets and can run on those devices. The main reason to choose the phone or tablet is that they are portable and can be carried anywhere and can give service in any rural areas of the countries. We also choose an Android device because of its worldwide availability and low-cost device and also it is easier for application development in Android platform. We developed eHealth app which principal user is a health worker and assisted them for recording patient detail electronically instead of paper and pen format through the patient form. It also assists them for the collection of the vital sign parameters automatically in the app from different PHD devices instead of manually recording in vital recording form. The application also has a function to analyze the vital life sign which provides with a score which helps the health worker to decide which level of care does the patient needs. We can also record the family and medical history of the patient. This application also provides an option to examine a patient based on symptoms provided by them and show the related information what can the patient do further, remedy information, when to see the doctor, overview of disease etc.

2. How the application can read the data from medical sensors automatically and can reduce human input error?

   We built a programme in our application that it can read the data from the medical sensor devices and can display and save automatically in our app. Our medical sensor devices are equipped with Bluetooth and BLE so as our mobile devices so this transport technology is used for data exchange between the medical device and mobile device. The medical devices are attached to the body of the patient, and the readings are taken then after readings are taken devices broadcast the data which are then caught and parse by the application and displayed on the screen and copied automatically in respective vitals form field which reduces the input error.

3. How can we make ease of use of the application by primary health workers with basic health knowledge or with little training?

   We designed the UI of the app simple and elegant so that it will be easier to operate by the primary health workers, its simple process they need to understand the English language and follow the instruction. They need a little training about the medical devices where they are attached on the patient body when to start the reading process on the app, how to save the data on the app and all these can be done in about a couple of minutes. A health worker can create their profile and right away start registering the patients, for measuring from devices they should click on measure option on necessary form and screen will show up where they can read data from several devices. Also, the score is calculated automatically and show in the screen inside the patient summary with recommendations which will be a more comfortable view for the health worker. And for analysing symptom they can get a selection option just click and move forward according to answer got from patient and finally conclusion scree will appear.

4. How to make it useful in a low-bandwidth network or in offline mode too?

   The application which we build for now uses no use of network so it can be used entirely offline all the data are based locally in the database. All the patient information, vital readings, calculated score, and history are saved on the device itself and can be retrieved locally if need for later. Here symptom assessment is also performed without the requirement of the network.

6.3 Findings from Testing

While doing the testing of the app we put some questionnaire to find the result from the participants. We had tested with ten different people with a question related to our app, and the effect was carefully examined. It is necessary to mention that we had difficulty to get a significant result because of the limited time. The test result was based on the ten participants and their experience towards the app. We are unable to test the app with the real health worker and with real patients whose opinions and experience would have resulted into different consequences and helps to develop better apps according to their needs for a real-world scenario. So these result from the limited patient cannot be considered as objective and convincing.

Also, we found out that from the test that the user interface of the app is highly appreciated and satisfactory. Majority of the user said that the app was user-friendly and comfortable to use. This app succeeded in fulfilling the requirement of measuring the vital parameters from different devices which meet requirements for the users. Few people also had raised the question concerning the app for some improvements and adjustment. Some of the concerned raised are listed below and will be corrected on the upcoming update.

- Some of them experience the crash and freeze of the application during their use we will handle this issue in the upcoming update.
- Some of them raise the question about the possibility of adding more medical devices, and we said we could add as many types of medical devices found in the market as long as they comply Continua standard.
- Some of them are also concerned about the authenticity of reading obtain by the medical devices and answer for this is yes we used certifed devices for measurement.
- They also raised question about the involvement of the medical experts involved in designing this app, but for now we had not involve any medical expert but we had used all the standard which are medically certified and avaliable.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

The main aim of this thesis research articulated from the beginning was to develop an application which can run on smartphones or tablet devices which advances the patient examine procedure by the health worker in rural or remote areas of many countries. This will help to improve the healthcare standard of patients with the ability to get basic health services and also helps the health worker to keep track of the patient health conditions. The primary goal of this project is to collect the vital life sign parameters from the patient using the medical sensor devices and observe the patient condition by their calculated score also analyzing the symptoms of the patient and provide them with the recommendation and to collect medical and family history of the patient. Therefore, an android based mTeleHealth-UiA app was developed to address these situations related in remote or rural areas where there lack primary health services. The application was developed on the base of the framework we designed, and the wireframe of the apps was designed. Moreover, the implementation was done based on these design and our requirements, finally model app was being developed and tested. The app was tested on the final stage, and due to the time limit, it could not be tested in a real scenario with real health workers and patients. Due to the time limit, the app has been affected and not tested with the intended user, but the involvement of the participant was quite impressive. According to the participants we find out the user interface was friendly and easy to use the application was also able to perform its intended task. Most of them were quite satisfied with the use of the overall app.

7.2 Future Work

There are still many possibilities to extend this application in the near future and had great potential that it could benefit huge no of people not only in rural areas but also every part of the world. We below discuss the significant extension for this app in the future.

Medical expert involvement for diagnosis

Here in the application initially we had planned to add the doctor's login page and also created wire-frame for that too, but due to time limitation, we cannot add those things. So, we can create doctor login where he can see the information collected by health-worker and can give some recommendation instantly. For this process, we also need the use of basic network connectivity, so this network feature can be added in future because nowadays most of the area we can get connectivity due to the rapid advancement in the communication sector. With the help of the network, we can also initiate the live session between the patient and doctor with developing a new video calling feature on the application.

Integration of more sensors with app

We had only integrated three sensors for to measure vital life sign parameter in this application, but in the near future, we can add more medical devices so that we can collect more data from the patient which makes easier on diagnosing and overviewing the condition of the patient. We can integrate more sensor devices such as Blood sugar measuring devices, activity monitor, strength fitness devices any many others.

Mapping collected data to clinical standard format

With the help of the current application, we can collect information from the patient and save in our local database as a medical record, but this medical and clinical information can be mapped into different standard HL7 standards, SNOWMED CT, Logical Observation Identifiers Names and Codes (LOINC) and many others for interoperability according to our requirements.

Using AI to analyze symptoms

In this application, we used the knowledge base information saved in our database to provide information about their health condition by analyzing the symptom. However, in future, we can add a feature that uses AI and can perform to get a better result by analyzing all the patient information, history and could give a better result, observation and recommendation with more accuracy than the present feature.

Making available in store for public use

Now the application is for only custom use and cannot be found on the market store to download. So, we can make some improvement on our application and can make ready to place in application stores so that it can be downloaded by the public also and can use according to their needs.

Bibliography

[1] ANDROID DEVELOPER. https://developer.android.com/studio/intro. Accessed: 2019-03-25.

[2] CONTINUA DESIGN GUIDELINES. https://www.pchalliance.org/continua-design-guidelines. Accessed: 2019-02-11.

[3] CONTINUA OVERVIEW. https://members.pchalliance.org/document/dl/1225. Accessed: 2019-03-20.

[4] ISO/IEEE 11073-10418:2014. https://www.iso.org/standard/61897.html. Accessed: 2019-02-11.

[5] ISO/IEEE 11073 Personal Health Data. http://person.hst.aau.dk/ska/MIE2008/ParalleSessions/P 1530/Sta-30Clarke. Accessed: 2019-02-20.

[6] ITU-T Rec. H.811 PHD Interface. https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-H.811-201607. Accessed: 2019-03-25.

[7] JSON INTRO. https://www.w3schools.com/js/js_json_intro.asp. Accessed: 2019-04-25.

[8] Mobile and Tablet Operating System Market Share. http://gs.statcounter.com/os-market-share/mobile-tablet/worldwide/#monthly-201803-201902-bar. Accessed: 2019-03-25.

[9] Research2Guidance . https://Research2guidance.com. 2017. [2018-03-13]. 325,000 Mobile Health Apps Available in 2017. Accessed: 2019-03-25.

[10] WHO and World Bank . https://www.who.int/news-room/detail/13-12-2017-world-bank-and-who-half-the-world-lacks-access-to-essential-health-services-100-million-still-pushed-into-extreme-poverty-because-of-health-expenses. Accessed: 2019-02-11.

[11] Bluetooth, S. Bluetooth core specification v2. 1+ edr, 2007.

[12] Burney SMA, Mahmood N, A. Information and communication technology in healthcare management systems: Prospects for developing countries. *Int. J. Comput. Appl. 4*, 2 (2010), 27–32.

[13] Chin, A., Xin, X., and He, J. Exercise lowers blood pressure. *Annals of internal medicine 136* (2002), 493–503.

[14] Cook, V. E., Ellis, A. K., and Hildebrand, K. J. Mobile health applications in clinical practice: pearls, pitfalls, and key considerations. *Annals of Allergy, Asthma & Immunology 117*, 2 (2016), 143–149.

[15] Developers, A. Android studio. *Android Developer,[Online]. Available: http://developer. android. com/sdk/.[Accessed 25 March 2016]* (2014).

[16] Ehringer, D. The dalvik virtual machine architecture. *Techn. report (March 2010) 4*, 8 (2010).

[17] Hartig, H., Hohmuth, M., and Barns, P. System integration for the android operating system. *Proceedings of CollaborationCom 7* (2010), 73–88.

[18] Heger, D. A. Mobile devices-an introduction to the android operating environment design, architecture, and performance implications.

[19] Irving, G., Neves, A. L., Dambha-Miller, H., Oishi, A., Tagashira, H., Verho, A., and Holden, J. International variations in primary care physician consultation time: a systematic review of 67 countries. *BMJ open 7*, 10 (2017), e017902.

[20] Jara, A. J., Zamora, M. A., and Skarmeta, A. F. *Knowledge acquisition and management architecture for mobile and personal health environments based on the internet of things.* 2012.

[21] Jones, M. Newsdig: The national early warning score development and implementation group. *Clinical medicine 12*, 6 (2012), 501–503.

[22] Jose, A. D., and Collison, D. The normal range and determinants of the intrinsic heart rate in man. *Cardiovascular research 4*, 2 (1970), 160–167.

[23] King, J. R., and Farner, D. S. Energy metabolism, thermoregulation and body temperature. *Biology and comparative physiology of birds 2* (1961), 215–288.

[24] Kolic, I., Crane, S., McCartney, S., Perkins, Z., and Taylor, A. Factors affecting response to national early warning score (news). *Resuscitation 90* (2015), 85–90.

[25] Larson, R. S. A path to better-quality mhealth apps. *JMIR mHealth and uHealth 6*, 7 (2018).

[26] Martinez, S., Gerdes, M., Kumar, S., Kumar, A., Gill, S., and Loudon, G. ehealth activists: the lifeline for remote rural villages in india.

[27] Mosa, A. S. M., Yoo, I., and Sheets, L. A systematic review of healthcare applications for smartphones. *BMC medical informatics and decision making 12*, 1 (2012), 67.

[28] Park, K., and Pak, J. Implementation of a handheld compute engine for personal health devices. *International Journal of Smart Home 6*, 2 (2012), 59–64.

[29] Patel, B. K., Chapman, C. G., Luo, N., Woodruff, J. N., and Arora, V. M. Impact of mobile tablet computers on internal medicine resident efficiency. *Archives of internal medicine 172*, 5 (2012), 436–438.

[30] Peters, D. H., Garg, A., Bloom, G., Walker, D. G., Brieger, W. R., and Rahman, M. H. Poverty and access to health care in developing countries. *Annals of the New York Academy of Sciences 1136*, 1 (2008), 161–171.

[31] Peters, J. L. Vital signs monitoring apparatus, May 17 1983. US Patent 4,383,534.

[32] Prytherch, D. R., Smith, G. B., Schmidt, P. E., and Featherstone, P. I. Viewstowards a national early warning score for detecting adult inpatient deterioration. *Resuscitation 81*, 8 (2010), 932–937.

[33] PubMed. *The Lancet. Rural health inequities: Data and decisions. Lancet.* PubMed, 2015;385:1803.

[34] Smith, G. B., Prytherch, D. R., Meredith, P., Schmidt, P. E., and Featherstone, P. I. The ability of the national early warning score (news) to discriminate patients at risk of early cardiac arrest, unanticipated intensive care unit admission, and death. *Resuscitation 84*, 4 (2013), 465–470.

[35] Song, H.-S., and Lehrer, P. M. The effects of specific respiratory rates on heart rate and heart rate variability. *Applied psychophysiology and biofeedback 28*, 1 (2003), 13–23.

[36] Sucala, M., Cuijpers, P., Muench, F., Cardo, R., Soflau, R., Dobrean, A., Achimas-Cadariu, P., and David, D. Anxiety: There is an app for that. a systematic review of anxiety apps. *Depression and anxiety 34*, 6 (2017), 518–525.

[37] Van Dyk, L. A review of telehealth service implementation frameworks. *International journal of environmental research and public health 11*, 2 (2014), 1279–1298.

[38] Williams, B., Alberti, G., Ball, C., Bell, D., Binks, R., Durham, L., et al. National early warning score (news): standardising the assessment of acute-illness severity in the nhs. *London: The Royal College of Physicians* (2012).

Appendix

Due to the many number of lines of the codes I had uploaded the code file and package file apk of the application on the GitHub and Dropbox and link are as below.

Dropbox:
https://www.dropbox.com/sh/s8agrqewd0va5t0/AACvvKa9imu
dKVaXGpN9ADqva?dl=0

GitHub
https://github.com/pankajkhatiwada/Thesis-Application