Senior Project Report

The Design and Implementation of a Mobile Health Self-Management Application for Older Adults in China

Xiaoting Fu

0930663

Advisor: Dr. Changjiang Zhang

May 24, 2017

Department of Computer Science

Wenzhou-Kean University

Acknowledgement

致谢

Thanks to my parents for their love and support for me all the time. Thanks, Dr. Pinata Winoto, Dr. Changjiang Zhang for their generous support and wise instructions on me for this project. Also thanks to the 16 testers who voluntarily participate in the testing of this application, without you, I could not collect such valuable feedbacks.

感谢父母一直对我的爱和支持。 感谢 Pinata Winoto 博士，张长江博士对我的这个项目的支持和明智的指导。 此外，感谢 16 位测试人员自愿参与了此应用的测试，没有您，我无法收集到这些宝贵的反馈。

Table of Contents

1 Introduction

China is facing the problem of population aging. Increasing number of older adults are suffered from chronic health problems and need affordable solutions. With the universal adoption of mobile phones among Chinese older adults, it is potential to use mobile phones as technology-based intervention (TBI) tool for older adults. The goal of my application is to enable older adults to manage their own health condition, facilitate disease prevention and encourage a healthy lifestyle.

1.1 Background and Literature Review

Given the increasing number of aged people, China has entered the aging society with the largest population of older adults in the world. Statistics from China National Committee on Aging (CNCA) show that, by 2016, the population aged over 60 has reached 222 million, and it is predicted that by the end of 2020, the aged population in China will become 243 million [Wang 2016]. Older adults, who are the most vulnerable population are more prone to chronic diseases. A WHO report on aging and health condition in China shows that by 2013, around 50% (100 million) of older adults in China experienced chronic diseases and the prevalence of them increase with age [WHO 2015].

The cost of medical care for these diseases are substantially increased with the population growth of older adults. Hence it's urgent for the healthcare industry to find affordable and effective solutions to relieve the burden of both the cost and the required labors for medical care.

Compared to internet-delivered desktops and laptops based health services, mobile health services have the advantage of flexibility and greater ability for frequent interaction and communications unbounded by time and space [Deng 2014]. While there exist many systems constructed for clinical professionals and healthcare administration, recently there been an increasing popularity in applying the technologies in consumer health, empowering patients to take control and play a proactive role in managing their health [Jonathan 2013].

Recently, with the universal adoption of smartphone, China's mobile phone users has reached 563 million. Also, for older adults, a study has shown that the penetration rate of the mobile phone is much higher than the internet or desktop [Chen et. al. 2013]. Therefore, given the increasing adoption rate of mobile phones among older adults and their incline of using mobile phones, it is promising to utilize mobile phone applications to manage their health and positively improve their life quality and well-being.

1.2 Purpose

Despite the potential of delivery such healthcare management service to older adults, as a newly emerged tool, it will still face many obstacles in the way of delivery. Therefore, physical and psychological characteristics of older adults should be identified before the development of such system. Therefore, the encouraged design principles will be adopted (i.e. Heart and Kalderon's [Heart et.al. 2013]) in this study. The existing Chinese mobile health applications including e-doctor (i.e. Chunyu Yisheng), PA training (Keep, Mi Sports, Yuepaoquan) and etc., are all incapable of providing customized, older-adults-oriented self-management and self-report functionalities. Therefore, I find there is a need

for me to design and implement an Android application that is designed upon the characteristics of older adults and their needs, help them build and manage their health profile, and track their daily physical activities.

2 Proposed Plan

2.1 Objectives and outcomes

After understanding the current project background and the existing problems, the next crucial part is to transform the understanding of current problems into a specific solution to the problem. The careful design and detailed implementation specifications should be addressed to increase the feasibility of this project. The specified objectives are identified in 2.1.1, and the outcomes that should achieve are defined in section 2.1. 2.

2.1.1 Objectives

The objectives covering all stages of development are listed below, some technical details are eliminated due to the coverage of them will be presented in section 2.2.3.

A. Identify physical and psychological characteristics of older adults

B. Identify the specific needs on health management tools of older adults

C. Identify the detailed User Profile

D. Identify the specific medication or drug standard database that can be added to the system for user to search and easily record their medication

E. Identify the UI of the application based on design principles

F. Design the system according the A and B and the system must be supportive for customization

G. Implement the system according to E, F.

H. Evaluate the effectiveness of the system via alpha and beta testing

2.1.2 Outcomes

- A list of commonly-find physical and psychological characteristics of older adults are identified.
- A list of required functions for health self management for older adults are collected and incorporated into the system.
- A database is built to store user's personal information, the drug list, and physical activity by day.
- An Android application is built to achieve the required functions and some tentative functions. The system will display the physical activity level of user and they can manage their medication there.
- A usability analysis report is generated and the application is refined and adapted according to the result of the testing.

2.2 Project Methodologies

The project is divided into the following phases:

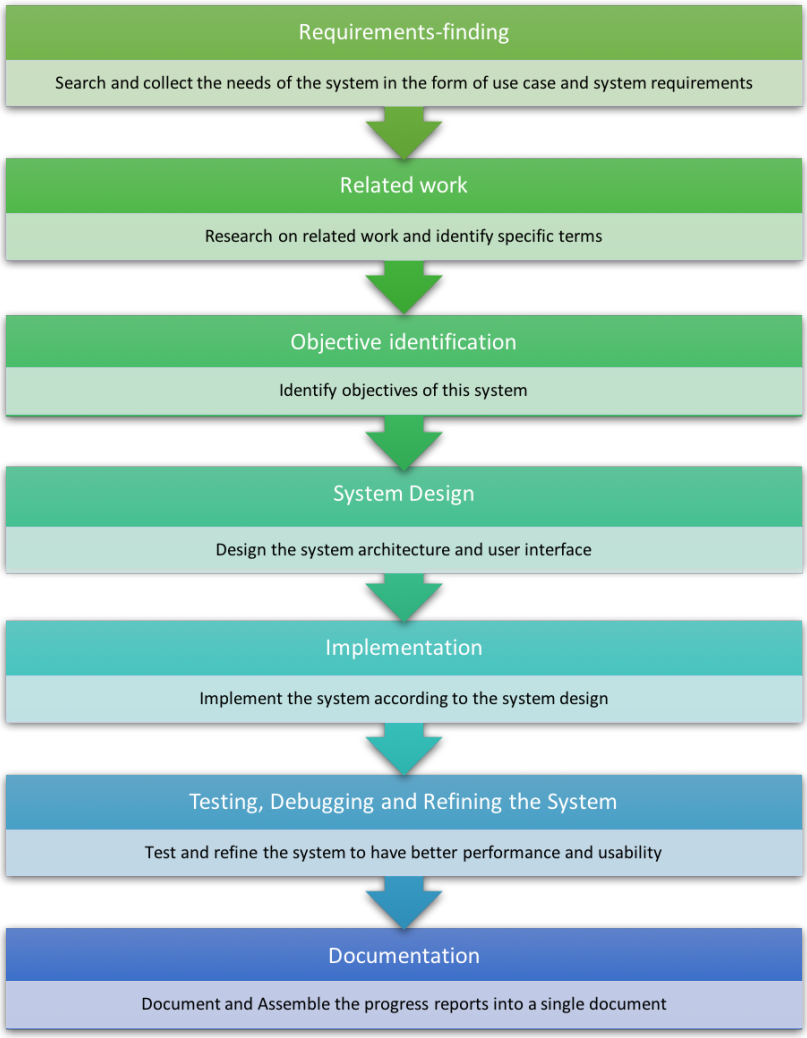


Figure 2-1 Project Phrase Diagram

2.2.1 Phrase 1-Requirements Finding

This part is listed as the initial stage of the project, as it is when developer searches for the possible topics and identify the need for developing the application. I find the need to develop an application for older adult due to my observation on the elder group and especially father's feedback of his interaction with their mobile phone. I observed that most of them carry a mobile phone for communication with their children, family, and friends. Even though some of them didn't use it very fluently, they like to carry them everywhere. As for my father, he is suffering from farsighted

eye problems, and his health condition is in a decline stage, daily he needs to take a significant amount of medicine according to his doctors' prescription. Sometimes, due to the significant amount, he may forget to take some drugs. Furthermore, I observed the same condition in my other family members (older than 55), whose health condition is declining and have the problem to manage their health problems efficiently. Based on my observation and discussion with them, I assumed that the mobile health self-management system might be beneficial for them. Wherever they are, whatever the internet connection condition is, they can always use a mobile phone to manage their health status, which usually runs an Android system.

2.2.2 Phrase 2-Related Works

After we identify our project idea, it's crucial to conduct a background study to learn about the current trends in this field, and discovering whether there is any similar project that can be the foundation for your study. Also, we can learn from techniques and evaluation process from others by reading the related work.

I have done the related study and presented them in the introduction part. Currently, I have read only 11 kinds of literature, but with the increasing identified searching key terms and their expanded keys that will be combined with the keys in a search attempt (Table 2-1), I will add more studies to the related works part and finally combine them into my final project report

Search Key Terms	Expanded Terms

“mobile phone”, “smartphone”	“adoption rate”, “health services”, “health application”
“older adults”	“physical characteristics”, “psychology traits”, “technology anxiety”, “behavior traits”
“self-management”, “self-reported”	-
“design principles”, “usability study”	“effectiveness”, “acceptance”

Table 2-1 Current List of Key Searching Terms with Expanded Terms

2.2.3 Phrase 3-Objective identification

Section 2.2.3.1 has summarized the objectives of the system eliminated the technical detail, which will be the primary focus of this section.

Use Case	Older Adult use the Mobile Application to Manage Health Condition
Actor	A older adult named Jack

Basic Flow	Jack is 66 years old; he suffers from visual decline, and chronic diseases such as high blood pressure, diabetes. His doctor has prescribed a list of medicines to him, and ask him to keep taking them daily, some medicine is twice a day, and some are three times a day, sometimes he will miss some medicine since he cannot remember, he finds it a problem to manage so many medicines without others to remind him. What's more, he wants to keep track of his health condition like weight, daily activity level (measured in steps), sleep time, every diet he has taken. The facts when collected together may give him a brief view of his recent health condition and become he can reflect on and sends to doctor if some disease occurs to him. Therefore, he wants some application to help him with the tracking of health condition and provide him with medicine alter and keep all his personal histories. Specifically, he can login to the system with a username and password, then if succeed, the system will display a major UI with his current steps, last night's total sleep time, the last diet he has recorded, and the medication record button, the TA tutorial button. When he clicks on each button, the system will direct him to the more detailed page of the data.

Table 2-2 Use Case on Older Adult Using the Mobile Application to Manage Health Condition

A study summarized some commonly-find functionalities of five existing mobile medication management applications [Grindrod 2014], including (1) Medication

list; (2) Reminder alarms; (3) Drug information; (4) Drug interactions; (5) Multiple user profiles; (6) Profile sharing via email and (7) Sharing across multiple devices. In my application, I will include (1), (2), (3), (5), (6) since they are the minimal requirement for the medication application.

Combined these selected summarized functions with the use case presented in Table 2-2, here is the functionality objective of this application.

A. Provide Login Function

B. Allow each user to customize their personal page, remove unwanted functions

C. Record and display Steps, can search for past records

D. Record and display Sleep time, can search for past records

E. Record and display Diet, can search for past records

F. Record and display Medicine as a list, set the alarm for the drug-taking reminder, can search for past records. The user can search for the medicine he wants to add from the drug database, and if not exist he can add his personal one, click each existing medicine it will display its detail information.

G. Provide PA training tutorial.

H. Provide the service to send the user's health profile to the desired email address

2.2.4 Phrase 4-System Design

System Overview

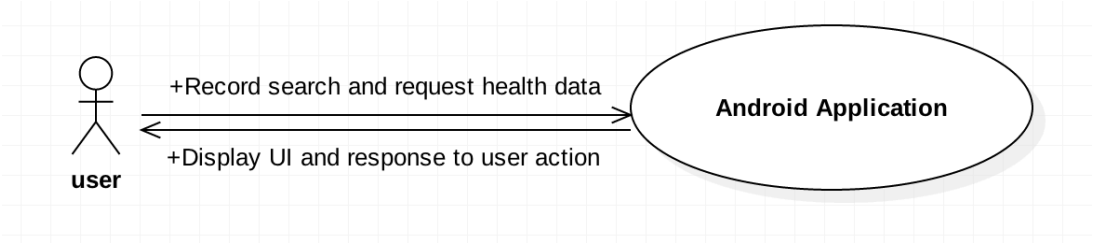


Figure 2-2 System overview

User Interface (UI) Design

The user interface design contains the principles that guided the design and the functionalities that should be achieved by each UI.

A. Design

Based on the research of Heart and Kalderon [Heart et. al. 2014], six design principles are encouraged (Table 2-3, when design the UI for my application I will adapt some principle based on my understanding.

Design Principles	**My Comments**
(1) factors reflecting the degree of usefulness;	Meet the need of them
(2) factors reflecting the degree of ease of use;	Simple, easy to use, adaptive
(3) technological issues;	With the minimal requirement of installation, easy to install and uninstall

(4) personal traits;	Build personal profile for each specific user and allow Personal customization of the app
(5) social issues;	Technology anxiety, communication needs
(6) facilitating issues	Larger fonts (customized fonts), audio response

Table 2-3 Mobile Healthcare Application Design Principles for Older adults

B. Functionality

Apart from the design, each UI will have its function that need to be achieved as is displayed in Table 2-4.

Name of Each UI	**Functionality Descriptions**
A Login UI	After successfully login, the user's profile will be displayed in the left side of the UI, displaying their username, height, weight, active days and other latest basic health conditions that is stored in the database.
The Main UI	Displaying the latest steps, sleep, the diet and the medication record, set the functionality they want to have (they can enable or disable the functions)

The Step Detail UI	Showing the detail of steps, displaying a broken line graph based on past 30 day's data and can search the record by day, the application will track the steps and automatic record it.
The Sleep Detail UI	Showing the detail of steps, displaying the last 10 days' sleep record and can search the record by day, if the user has a supportive wearable device, he can select the device and ask the device to send the data, if not the system will give him the template for him to record by himself
The Diet Detail UI	Showing the detail of diets in today and can search for past diet record by date, record the diet using the provided template
The Medication Record and Alert UI	The patient can add the medicine he need to take into his medicine list, he can select from existing medicine from the system, or if his medicine does not exist, he can type the name of the medicine and add the record by hand. He can set alter by select the times he need to take each medicine every day.

The PA Tutorials UI	This part is an optional one, the user can choose to not display this section by click the dismiss button, if he want to keep this section, it will provide the text-based tutorials on several common PA.

Table 2-4 UI names and Functionality Details

2.2.5 Phrase 5- Implementation

The implementation stage can begin when the design is finished. Three critical issues about implementation will be addressed here, including the input, output and implementation programming language.

System Input

The system input comes from the system user's recording, selecting and interacting with the system.

System output

This system can display statistics and graph to user and also send output in the form of web from via email to the target receiver.

Programming Language

The programming languages for this project, including Java.

2.2.6 Phrase 6- Testing, debugging and refining the system

After successfully implemented the android application, to enable the system to become runnable, complete, and usable I will conduct three types of testing. First,

Unit testing will be performed to fix the initial bugs in the system and the universal usage of this application in devices with different screen size, using Android Studio's JUnit. Second, system integration testing will be performed to test the completeness of the system's functions. In this testing, several testing scenarios will be designed and delivered to users to discover any additional unmet requirements. Third, usability testing on the ease of use of this application and the adherence of the implementation compared to the proposed design. After the testing, I will collect test results and make adjustments to the system, its UI, fixed the bugs and any functionalities missing.

2.2.7 Phrase 7- Documentation

After the final implementation, this system will be ready for show case. Therefore, all the documents written in the process of the development, including proposal, system requirement, related works, a progress report will be combined to form my final report. The final report will also include the analysis of the application, discussion of the future development goal and its limitations.

2.3 Project Schedule

Since this is a roughly three-month project, to ensure the quality and completeness of this project, the detailed timeline of this projected is listed in the Table 2-5.

WBS	Progress	Task Name	Start Time	End Time	Duration
1	80%	Related Literature Reading	2017/02/13	2017/02/15	3
2	100%	Prepare proposal	2017/02/16	2017/02/20	4
3	0%	Prepare System Requirement Specification	2017/02/20	2017/02/26	7
4	0%	Prepare Design Specification	2017/02/24	2017/02/27	4
5	0%	Implementation	2017/02/27	2017/03/30	31
6	0%	Consultation	2017/03/30	2017/03/30	1

7	0%	Midterm Presentation	2017/04/10	2017/04/10	1
8	0%	System Testing	2017/03/31	2017/04/02	3
9	0%	User experience testing	2017/04/02	2017/04/16	15
10	0%	Prepare testing document	2017/04/17	2017/04/24	8
11	0%	Prepare final report	2017/04/24	2017/05/25	31
12	0%	Prepare system-in-action video	2017/05/01	2017/05/07	7
13	0%	Show-case	2017/05/10	2017/05/10	1
14	0%	Final presentation	2017/05/11	2017/05/11	1
15	0%	Final report due	2017/5/21	2017/5/21	1

Table 2-5 Grants Chat of Project Schedule

2.4 Resource Estimation

2.4.1 Hardware Resource

Laptop (for software development)	OS: macOS Sierra; CPU: 2.4 GHz Intel Core i5; RAM: 8GB 1600 MHz DDR3; Graphic: Intel Iris 1536MB
Ali Yun Server	1 core/ 2G; 1M bandwidth; 20G cloud storage
Android Phone	Android 5.1 OS; 5.5 inch; 2GB RAM; 32GB storage
USB	USB 2.0/3.0 to micro USB line

2.4.2 Software Resource

Operating System	macOS Sierra
Development Tool	Android Studio, XAMPP
Database	MySQL
SDK	Java SDK, Android SDK

API	Mi Health API

3 Software/System Requirements

3.1 Introduction

3.1.1 Purpose

The purpose of this section is to provide software requirements specification for the mobile health management system for older adults, “O Health”.

3.1.2 Intended Audience and Reading Suggestions

This document is intended for developers of this mobile application; this document provides specific requirements that guide their development. Therefore, developers should read this document carefully before the actual implementation of the software application.

3.1.3 Scope

The “O Health” is a comprehensive mobile health management application, which helps older adults to record their physical activity, health status (including steps, sleep, and diet), and manage their medication records. Users of the “O Health” application will benefit from this application by gaining a better knowledge of their health condition, so that they can provide detailed personal health data to clinician if disease occurs and help disease diagnosis, or they will benefit from the medication management function of this application which will help them to manage their drug taking every day. The mobile platform of this application is Android, and the database it used is the NoSQL database provided by Firebase. Since the users of this application is a group of people with distinctive characters, the design of this application will mainly take care of their needs.

3.1.4 References

Firebase (2017). https://firebase.google.com

Google Inc. (2016). User Interface Guidelines. https://developer.android.com/design/index.html

McDonnell, C., Werner, K., & Wendel, L. (2010). Electronic health record usability: Vendor practices and perspectives. AHRQ publication, (09), 10.

IEEE Software Engineering Standards Committee, "IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications", October 20, 1998.

Meingast, M., Roosta, T., & Sastry, S. (2006, August). Security and privacy issues with health care information technology. In Engineering in Medicine and Biology Society, 2006. EMBS'06. 28th Annual International Conference of the IEEE (pp. 5453-5458). IEEE.

3.2 Overall Description

This section will give an overview of the whole system. The system will be explained in its context to show how the system interacts with other systems and introduce the basic functionality of it. It will also describe what type of users that will use the system and what functionality is available for them. At last, the constraints and assumptions for the system will be presented.

3.2.1 Product Perspective

The application is a self-contained software, independent of other larger system. It is made up of three parts: the mobile software, the wearable device, and the cloud server

along with the database service it provided. The Android application will be able to support multiple users, helping them collecting and recording their personal health data.

The Android application is the central part of the system with Firebase API included in the system, see Figure 3-1. The Firebase API provides data communication (user authentication, data retrieval, and data storage) between the Android application and the Firebase real-time database, which is located in the Firebase cloud server. The Firebase is a platform providing mobile and web application developing tools, including application data analytics, user authentication, real-time database (data are stored in Firebase Cloud) based on REST API, file storage (including upload and download services) and Android Application Test Lab. Since the application is only in the initial stage, I will use the Spark Free plan of Firebase, which only allowing 100 simultaneous connections, 1 GB database storage, 10GB/month download. The Firebase database is a NoSQL database.

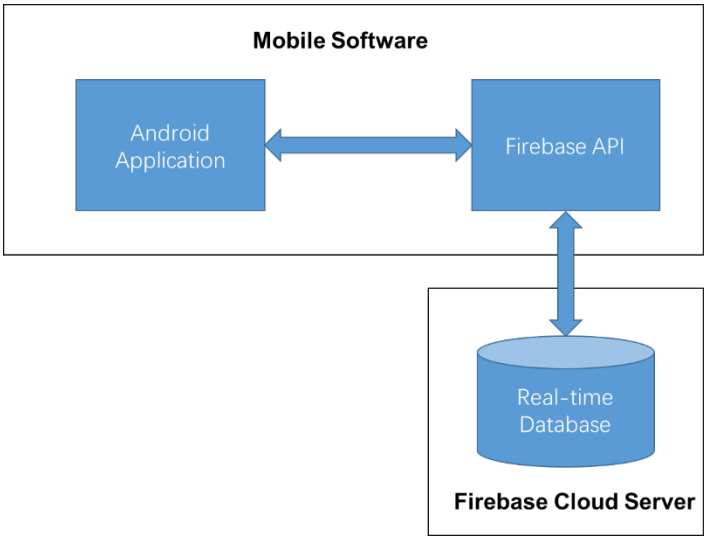


Figure 3-1 System Architecture Block Diagram

3.2.2 Product Functions

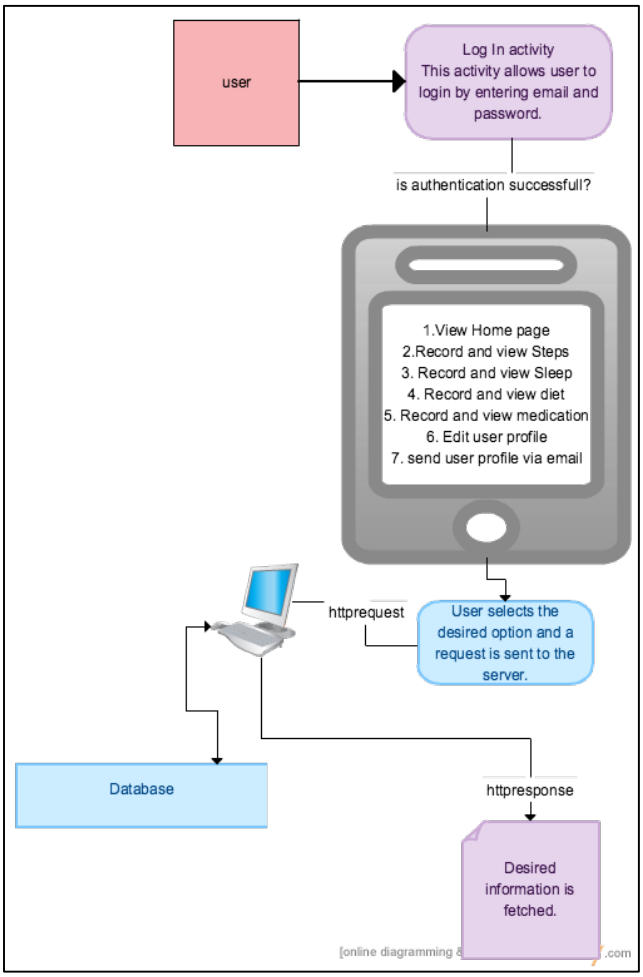


Figure 3-2 Data Flow Diagram of Android application "O Health"

The major functions O Health provide will include user profile synchronization through different devices, user step, sleep, diet, and medication recording. The device of the data flow in this application is shown in the data flow diagram, see Figure 3-2 (the detailed description of functionality will be shown in section 3.1).

3.2.3 User Classes and Characteristics

There is only one type of user that will interact with this system—the older adults. In this section, I will evaluate the unique characteristics of them.

The older adults have the unique characteristics that should be taken care of; it will be discussed in the design specification document.

3.2.4 Operating Environment

The software will operate in Android OS 5.0+, the hardware will be the android phone and the android applications in the phone can all coexist peacefully with this software.

3.2.5 Design and Implementation Constraints

As is mentioned in section above the software is an application developed specifically for the Android operating system. Therefore, the user must have an Android phone to run our software. Also, the minimal Android OS version is 5.0. Also, since the application will allow user to save their data in cloud database which is based on Google's service, therefore the user should have Google play's service installed on the phone and the internet connect should be able to access Google's service, i.e. in China, the user should connect to VPN service that allows them to access Google's service.

3.2.6 User Documentation

The documentation that designed to help user will be a user manual in the form of HTML, which will be displayed in the "Help" section in the application. The user manual will be delivered by this application to teach the user how to use this application and help them make the most the application.

3.2.7 Assumptions and Dependencies

One of the assumptions about the application is that the mobile phone is based on Android system and the OS is Android 5.0+ with enough performance and storage. If the phone does not have enough hardware resources available for the application, for example, the

users might have allocated them with other applications, there may be scenarios where the application does not work as intended or even at all. Another assumption is the mobile phone has an internet connection which allowing them to access Google Play Services. Without Bluetooth connection, the application cannot collect their data automatically, the user should input their data manually. Without the Internet Connect and Google play service, the user cannot login to the system and synchronize their data.

3.3 External Interface Requirements

1.1.1 User Interfaces

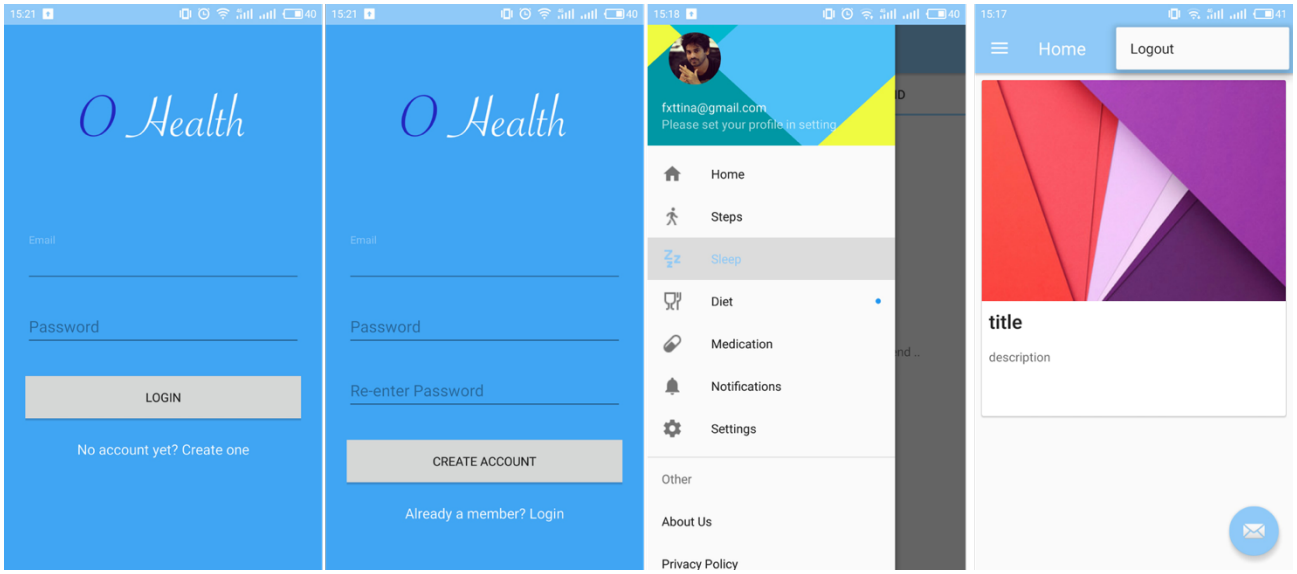


Figure 3-3 UI Framework (From left to right: Login UI, Signup UI, Main UI, and Home Fragment UI including a Logout menu)

The user interface follows the design idea of material design published in 2015's Google I/O conference, with the icon selected from the material IO's library and the design and the layout design following the drawer layout from material design. Figure 3-3 shows four screenshots of the sample UI, which is an unfinished framework, with the signup and login user interface alter user to create an account and use their registered account to access the

system. After the user login to the system, their profile photo and account id will be displayed in the header section of the drawer layout. The navigation menu in the Main UI will display a list of fragment items which will have their separate functionality to perform in fragment layout. Also, the user can choose to logout, using the logout button displaying in the toolbar of the home fragment, so that they can switch the account. During the user registration process error message will show up to inform the user to input correct form of email and strong enough password.

3.3.1 Hardware Interfaces

Since the mobile application doesn't have any designated hardware, it does not have any direct hardware interfaces. The physical Bluetooth and wireless connection device in the mobile phone are connected to hardware application programming interfaces (API) therefore and are managed by the underlying operating system in the mobile phone. The mobile phone connection to the hardware database is managed by Firebase web hosting's services.

3.3.2 Software Interfaces

The mobile application communicates through internet to access the Firebase Real-time database and the functionalities required including data storage, retrieval, and update.

3.3.3 Communications Interfaces

The application is based on cloud-database, so internet connection is essential for the data transportation from the application to the database. The communication standards it used in the network is HTTP. As for the communication among different system components, they are handled by the operating system in the mobile phone.

3.4 Functional Requirements Specification

3.4.1 Functional Requirements

3.4.1.1 User Registration and Login

The system shall allow the user to register by entering email and password.

The system shall authenticate user credentials and allow him to access the system when succeed.

The system shall allow the user to edit and update the profile information.

The system shall allow the user to verify the email through a verification email.

The system shall allow the user to logout the system.

3.4.1.2 User Health Profile Synchronization

The system shall synchronize the user's profile through different device with the same account.

3.4.1.3 Walking Steps Counting

The system shall allow the user to view his walking steps by date.

The system shall allow the user to record his steps manually if his device cannot collect his steps information automatically.

3.4.1.4 Sleep Time Recording

The system shall allow the user to view his wake up time and bedtime by date.

The system shall allow the user to record his sleep time manually.

3.4.1.5 Diet Information Recording

The system shall allow the user to search for a specific food by its name in a food database.

The system shall allow the user to add a new food to the database (using a food information template).

The system shall allow the user to select the food and record his diet information including a specific date, diet time (breakfast, lunch, dinner and snacks), food amount.

The system shall allow the user to view his diet history by date and diet time.

3.4.1.6 Medication Alert

The system shall allow the user to search for a specific drug by its name from a drug database.

The system shall allow the user to add a new drug to the database (using a drug information template).

The system shall allow the user to select the drug and record his medication taking information including an interval of taking, schedule and dose time, start time, duration and instruction, whether the user wants an alter from the system.

The system shall allow the user to view his medication taking history by date.

3.4.1.7 Sending Health Profile

The system shall allow the user to send his medical profile to a designated email address.

3.4.2 Use Cases

In this section, use cases will specify the detailed system specification.

The user has the following sets of use cases:

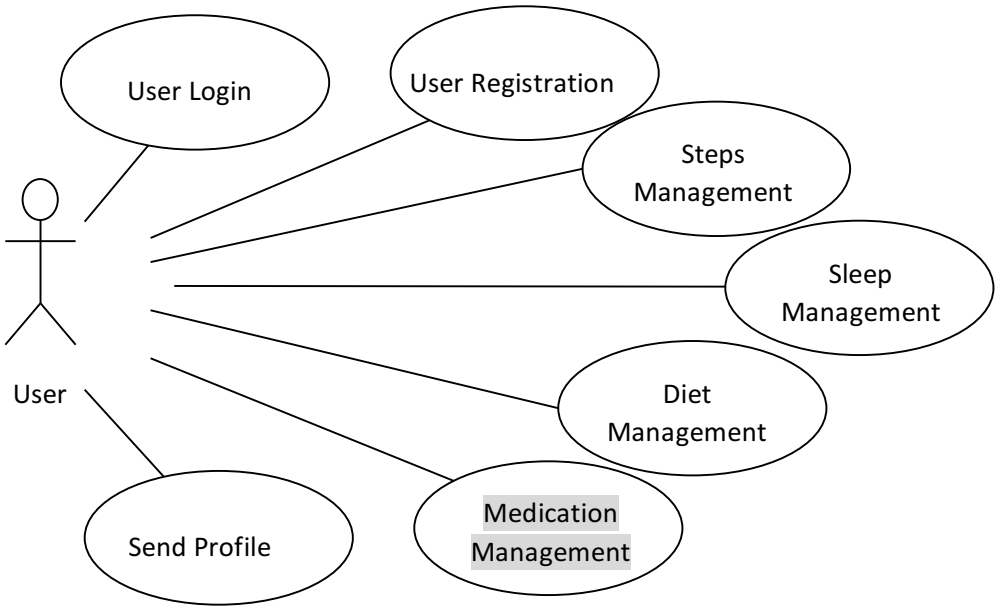
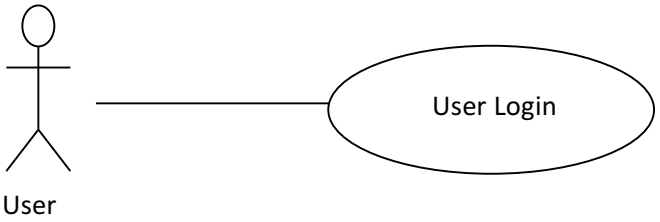


Figure 3-4 Use Case Overview

3.4.2.1 Use Case: User Login

Diagram:



Brief Description

The user opens the OHealth android application, login to the system by entering the correct email and password.

Initial Step-By-Step Description

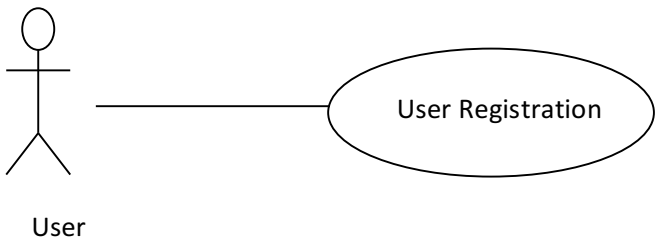
Before this use case can be initiated, the user has already accessed the OHealth Android Application.

1. The system displays an email and password Editbox prompting user to enter their credentials.

2. The user enters his email and password and click the “login” button.
3. The system authenticates the user’s credentials and displaying the “Authenticating…” progress dialogue prompting user to wait for the process.
4. The system displays the home page and inform user that he login successfully.
5. The system displays a “login failed” message, informing user to click a link displays below the login button with the text “not have an account, create one” and direct user to the user registration page.

3.4.2.2 Use Case: User Registration

Diagram:



Brief Description

The user register for a new account for the system.

Initial Step-By-Step Description

Before this use case can be initiated, the user has already accessed User Login page.

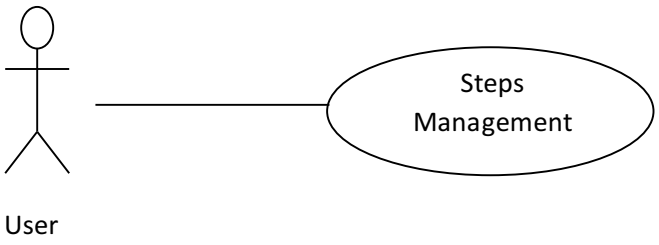
1. The system displaying the email and password Editbox, prompting user to enter his information.
2. The user enters an email and his password and click the “signup” button.
3. The system displays the “create account” process dialogue message while validating the data in the “email” and “password” Editbox to

ensure whether the former is an email address and whether the latter one align with the password standard this system requires.

4. The system saved the user's data into database.
5. The user clicks the "already have an account, please login" link to direct him back to the login page

3.4.2.3 Use Case: Steps Management

Diagram:



Brief Description

The user manages his walking data in the "Steps" page.

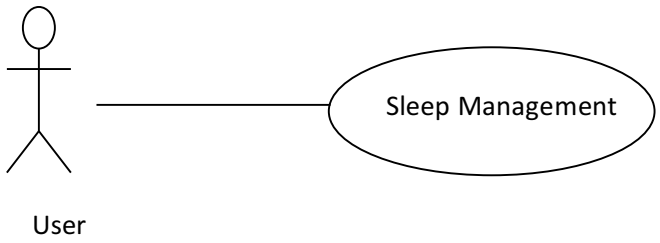
Initial Step-By-Step Description

Before this use case can be initiated, the user has already login to the system.

1. The system displays a menu including "Steps" as an option of that menu.
2. The user clicks the option button and the system direct him to the "Step" page.
3. The system displays the real-time steps taken by the user.

3.4.2.4 Use Case: Sleep Management

Diagram:



Brief Description

The user manages his sleep data in the "Sleep" page.

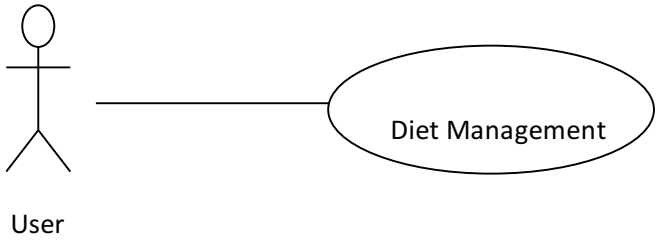
Initial Step-By-Step Description

Before this use case can be initiated, the user has already login to the system.

1. The system displays the length of sleep, start time, end time that user takes last time.
2. The user chooses to record his sleep by clicking the "Add Sleep" button.
3. The system directs the user to the "Add Sleep" page.
4. The user selects the start and end date and time for his sleep, and click the "save" button to save the sleep data.
5. The user presses backward button in the navigation bar.
6. The system directs the user to the Steps page and displays the sleep log by date the user selected.
7. The user can edit his existing sleep record and delete the unwanted one.

3.4.2.5 Use Case: Diet Management

Diagram:



Brief Description

The user manages his diet data in the “Diet” page.

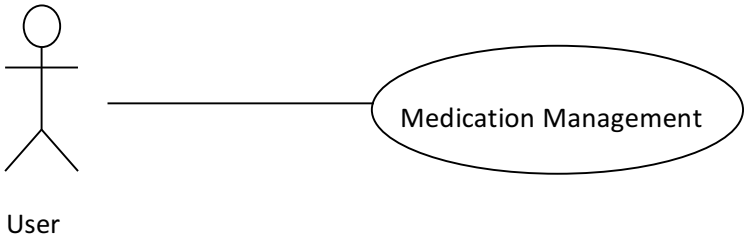
Initial Step-By-Step Description

Before this use case can be initiated, the user has already login to the system.

1. The system displays the total calories the user takes for the day.
2. The user clicks the “Add food” button to record his food.
3. The system directs the user to the “Add Food” page.
4. The user fill in the page and save the data.
5. The system displays the last added record in the “Diet” page.

3.4.2.6 Use Case: Medication Management

Diagram:



Brief Description

The user manages his medication data in the “Medication” page.

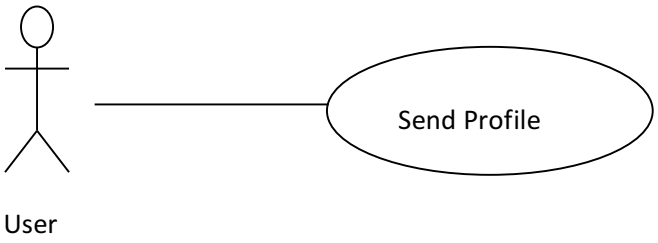
Initial Step-By-Step Description

Before this use case can be initiated, the user has already login to the system.

1. The system displays the medications that the user has added in the "Medication" page
2. The user clicks the "Add" tab to add new medication.
3. The system directs the user to the "Add medication" page.
4. The user fill in the template and click "save" button to save the medication information.
5. The system displays the newly added medication in the "Medication" page

3.4.2.7 Use Case: Send Profile

Diagram:



Brief Description

The user sends his profile to a designated email address.

Initial Step-By-Step Description

Before this use case can be initiated, the user has already login to the system.

1. The user selects the "Profile" page.
2. The system displays the user's profile in the page.

3. The user chooses the data he wants to send by enabling the switchers in the page and enter an email address.
4. The system assembling the selected data from the user and display the content to the user.
5. The user clicks the “send” button.

The system sends the profile as a HTML page to a designated email address.

3.5 Non-Functional Requirements

3.5.1 Security Issues in Health Information System

The security issues that should be considered are the security of user’s data, the account’s security and the privacy policy this application can provide to the user.

3.5.1.1 Access Control

First of all, the application required user to create an account and login to their account. With the system will restrain user to input only strong password to ensure their account security. Also, the authentication system is provided by Firebase Authentication method, which is till now very reliable.

3.5.1.2 Data Encryption

The user’s password, once saved to the database will be encrypted and even the database administrator cannot view the user’s data. This will secure the password and ensure the safety of the account.

3.5.1.3 User Privacy Policy

Within the app, the privacy policy tab will be available to the user where the application developer will declare that the user’s data will not be distributed in any ways.

4 Design Specifications

4.1 Introduction

The purpose of this document is to convey the design idea and architecture of the "O Health" application. This document defines how the developers intend to implement the application to function according to the software requirements previously submitted. The application is intended to provide useful functions for older adults to manage their daily health condition.

4.1.1 Goals and Objectives

The goal of the O Health development team is to design and develop a health-care management system for the mobile android platform. The application will allow the user to track their daily health activities (steps, sleep and diet) and create alter for the medication.

4.1.2 Statement of Scope

In O Health, user can choose to use any of the functions that the software provides for them, "steps", "sleep", "diet" and "medication" are integrated in one single application. It is a mobile Android OS-based healthcare application. The application will be cloud-based with the user authentication and user data stored conducted in the real-time cloud database.

4.1.3 Software Context

This application is being developed to release to any people who need it. With an email request, the APK of this application will be sent to the user for use.

4.1.4 References

Heart, T., & Kalderon, E. (2013). Older adults: are they ready to adopt health-related ICT? International journal of medical informatics, 82(11), e209-e231.

4.1.5 Major Constraints

Due to the unique nature of this project, an academic implementation of a health management tool, this software will have several uncontrollable constraints.

1. Project Size: The project will only have several selected important features which is most required for the intended audience for this project. Due to the time limitation, not all required feature can be implemented by this project. However, future expansion of this project will allow the developer to add more useful feature to this application. This project is designed to be easily expandable and will be easy to develop and add functions.

2. Project Timeline: The project deadline is constrained into one semester, and managed by the course schedule, therefore, only limited functionalities can be implemented. However, a project schedule is proposed in the proposal document, to constrain the features that can be achieved before the deadline.

4.2 Database Design

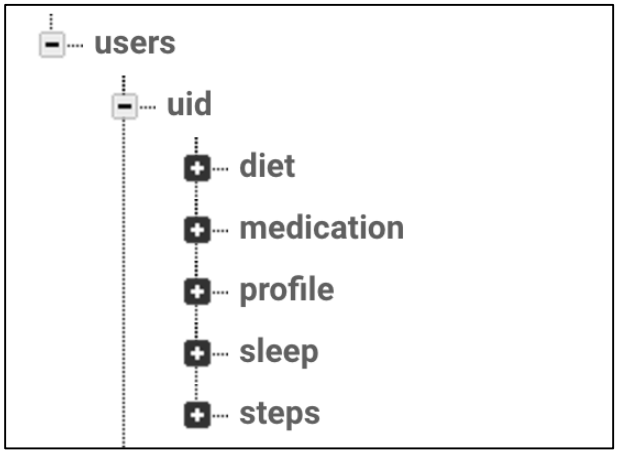


Figure 4-1 Database Structure Diagram

The O Health application hold its data in a NoSQL Firebase database. The data in NoSQL database are actually in JSON format, due to the mutually-convertible feature between JSON object and java object, the data can be defined by the java class and retrieve the java object to store it into Firebase database. Unlike SQL database, the NoSQL database has a more flexible structure whose structure are tree-oriented. Therefore, the main purpose of this section is to demonstrate the Java class that will be used to build the database's content.

Below is the structure of the database. "Users" is the root tree, with a child "uid" (which is used to sparated different users), the "uid" also has five child named "diet", "medication", "profile", "sleep" and "steps" each hold the user's data object for each kind.

The java classes above represent object that can be added into the database. Take the sleep java class for example, when added into database, the name of each java object attribute are present in the name part, like "numerate", and the value is in the follow, like "5". In this case, a java object attribute become a corresponding JSON object's name and the attribute's value is stored in the value part of that child.

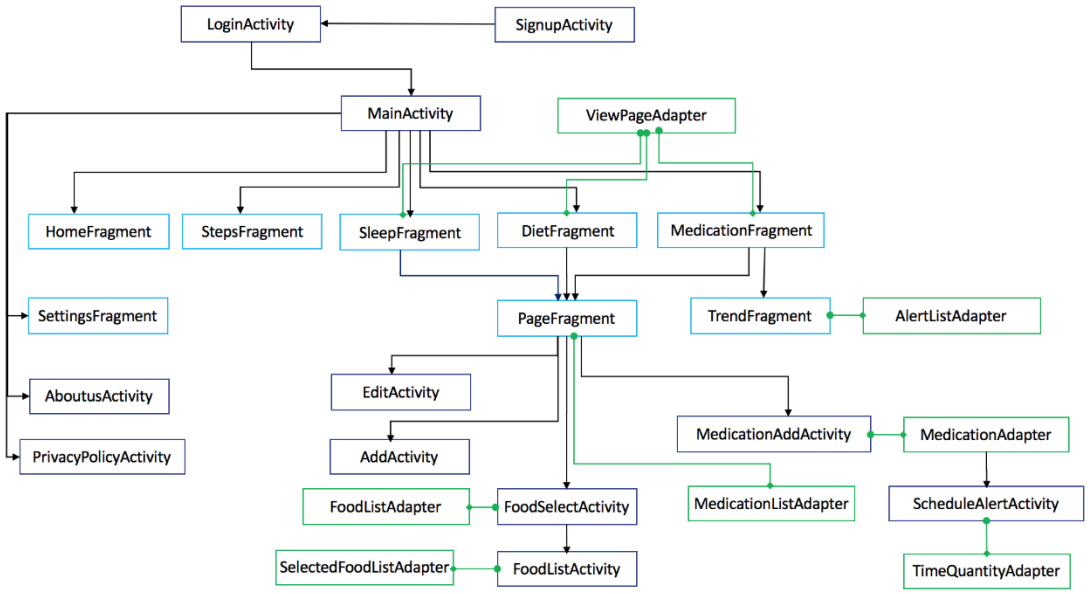
4.3 Architectural and Component-Level Design

The Android Project O Health will be built with the Google Android SDK for Java. It will be implemented with Java and XML. The project includes Android activities and fragments, with activities perform the interaction with data and fragment perform the presentation of data.

4.3.1 System Structure

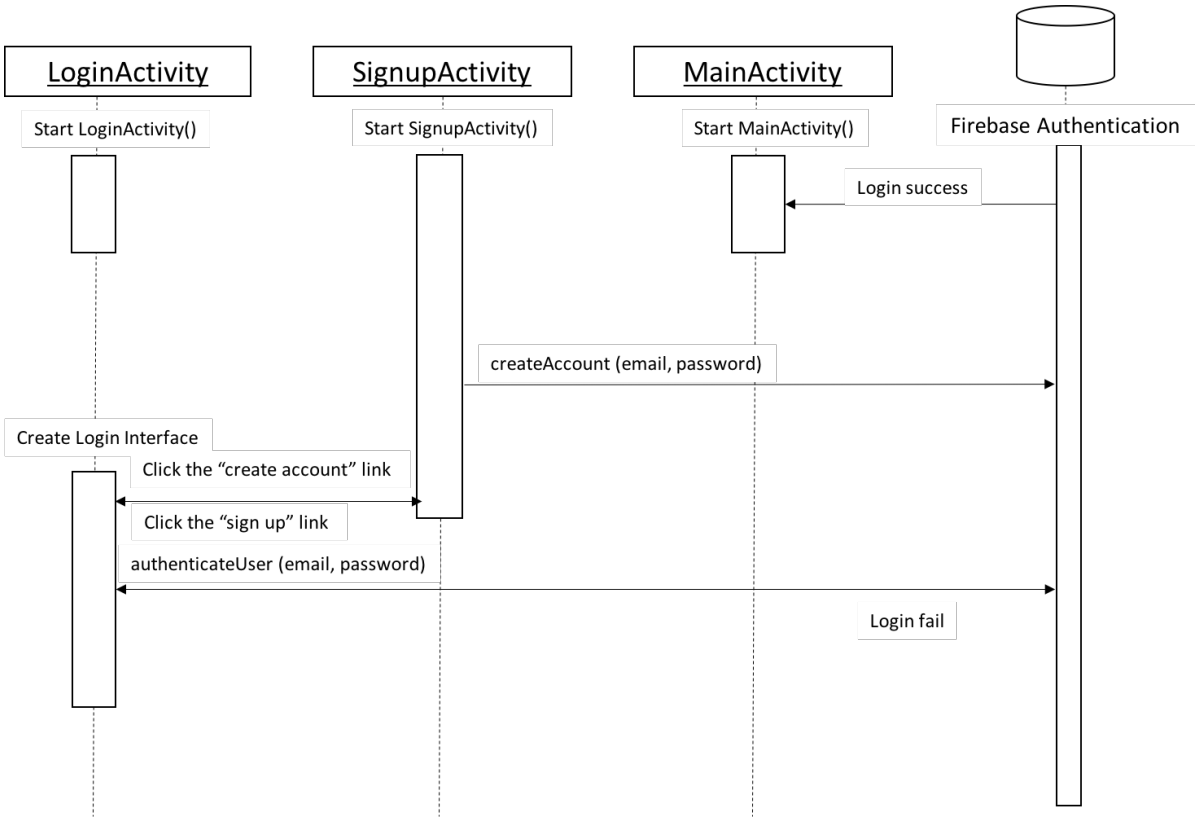
The application will be implemented using Android studio, and android SDK for Java.

4.3.2 Architecture Diagram



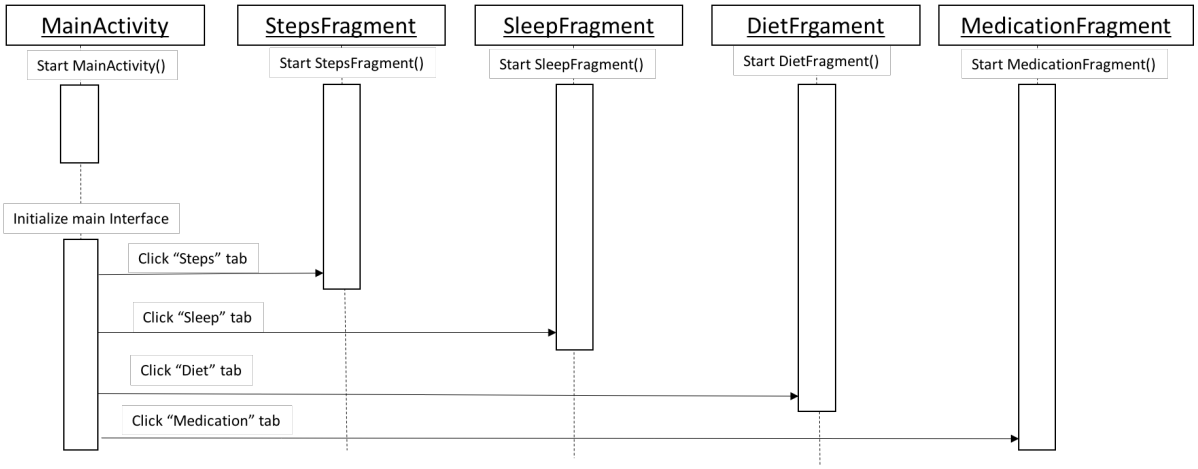
4.3.3 Sequence Diagrams

4.3.3.1 Application Launch Sequence



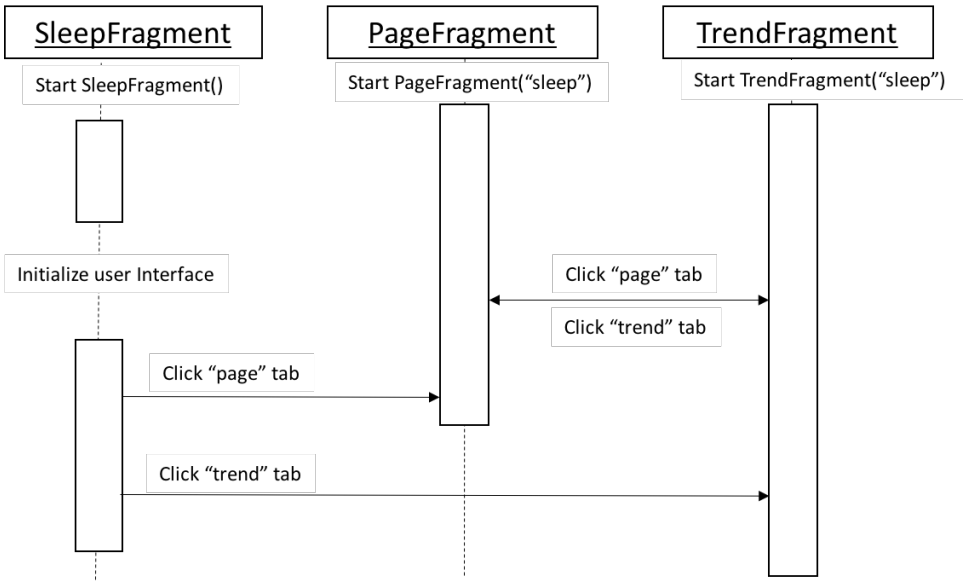
This diagram describes the activities performed when user launch the application. As is shown in the diagram, the “launch activity” is LoginActivity, which ask user to enter its email and password. If the user doesn’t have an account, the user can click “create account” link and the user will be directed to the “SignupActivity”, when account is created, the user can go back to the “LoginActivity”. If the user’s authentication succeeds, the system will direct the user to the “MainActivity”.

4.3.3.2 Application Built Sequence



This diagram describes the sequence after user has successfully login to the system. The MainActivity with its UI will be presented to the user, the user can choose from four tabs in the navigation menu: “Steps”, “Sleep”, “Diet”, and “Medication”, with each tab selected, the system will direct the user to a corresponding fragment.

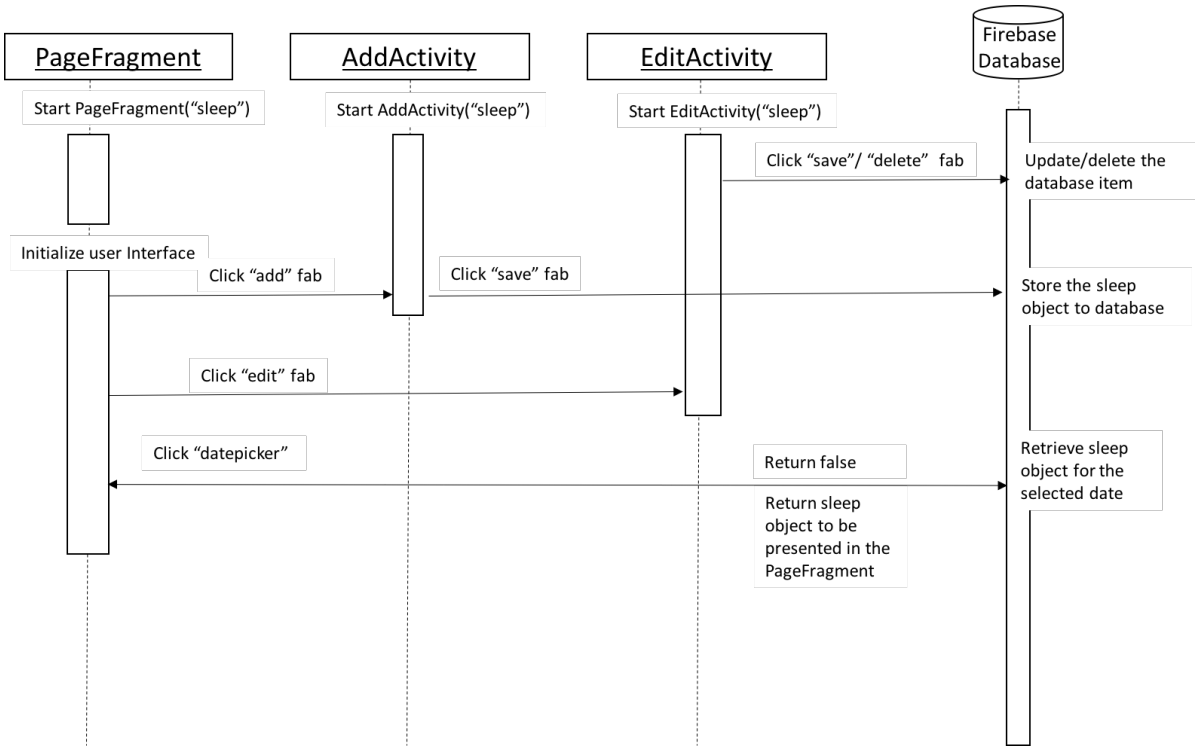
4.3.3.3 SleepFragment Switch Sequence



This diagram describes the sequence when user click the “Sleep” tab in the menu (the sequence for other three tab will be same with this, so we only take the “Sleep” tab for example). The SleepFragment holding a tablayout is actually a container of two other tab

fragments: “PageFragment” and “TrendFragment”. The user can switch between these two tab fragments using the tab in the layout.

4.3.3.4 Database Interaction Sequence



This diagram describes the sequence when user choose the PageFragment’s tab in the SleepFragment. The PageFragment with a UI specified for “sleep” will be presented to user. The user can select a date from the datepicker to see if sleep data has already record for that specific date. If there is no sleep data for that date, the user will see a “add” fab, the user can click the “add” fab to enter the “AddActivity” where he will set his sleep data and click the “save” button to save the data to the database.

However, if there is sleep data for that date, the Firebase database will return that sleep object to the PageFragment and the sleep hour will be displayed. The fab will change from “add” image to a “edit” image, informing user to alter the data instead of add new data.

When user click the fab, the “EditActivity” will be started displaying the old data that user has set the last time. If the user performs some changes and click the “save” button, the data in the database will be updated, or the user can click the “delete” button in the toolbar to delete the sleep data for that date.

4.4 User Interface Design

4.4.1 Descriptions of User Interface

The user interface will be split up into 5 major activities and three major fragments: LoginActivity, SignupActivity, MainActivity, SleepFragment, PageFragment, TrendFragment, AddActivity, and EditActivity. The user will be able to navigate through these screens to reach the components of the application that requires the user’s input.

4.4.2 User Interface Design Principles

Based on the research of Heart and Kalderon, several design principles for designing mobile applications for older adults are encouraged and will be consider for the design of interface:

1. Factors reflecting the degree of usefulness;

Since older adults are characterized by technological anxiety, a simple problem occurs from the application will make them stop using this application. Therefore, the core design of he application is its simplicity, only represent the desired data to the user and the unnecessary data will not be presented. For example, in the PageFragment for sleep, instead of presenting all recorded log from the database, the system only presents the data for a selected date. Moreover, the presented data will only include the sleep duration, which is the most important feature of sleep, instead of presenting all the sleep data.

2． Factors reflecting the degree of ease of use;

As the frequency of use increases, so do the user's desires to reduce the number of interactions and to increase the pace of interaction. After the user login to the system, the system will keep user login, instead of asking user to log in when the user the app for another time in the date. This feature will reduce user’s task and increase the application’s usefulness. Also to prevent user to type, the app will ask the user to select a date. In general, choices will be presented to the user if possible instead of asking the user to type in the data for themselves.

3． Technological issues

The older adults will resist adapting to the latest technology, therefore, the application should not expect the user to solve the application’s bug by themselves. Therefore, the application should test more cases to reduce the occurrence of bug.

5 Implementation

5.1 API employed

In this application, many open source API are employed in the android’s buile.gradle file. The table below lists the employed API and the role they play in this application.

Name	Source	Function
Show/Hide Password EditText	com.github.scottyab:showhidepasswordedittext:0.8	In the “Login” and “Register” page the API is used to display or hide password by clicking an “eye” icon.

	https://github.com/scottyab/showhidepasswordedittext	
MaterialSearchView	com.miguelcatalan:materialsearchview:1.4.0 https://github.com/MiguelCatalan/MaterialSearchView	In the “Food Select” page, the search view is displayed in the toolbar, the user can click “search” button and enter query text.
EazeGraph	com.github.j4velin.EazeGraph:EazeGraph:1.0.2 https://github.com/blackfizz/EazeGraph	In “Steps” page, the EazeGraph API is used to display the pie graph of steps and bar graph of past seven day’s historical steps.
Google Play Services API	com.google.android.gms:play-services-auth:10.2.0 https://developers.google.com/android/reference/packages	The application uses Google play services to perform login activity.
Material Rating Bar	me.zhanghai.android.materialratingbar:library:1.0.2	In “Sleep” page, the material rating bar API is used to display the rating bar with customized color and style.

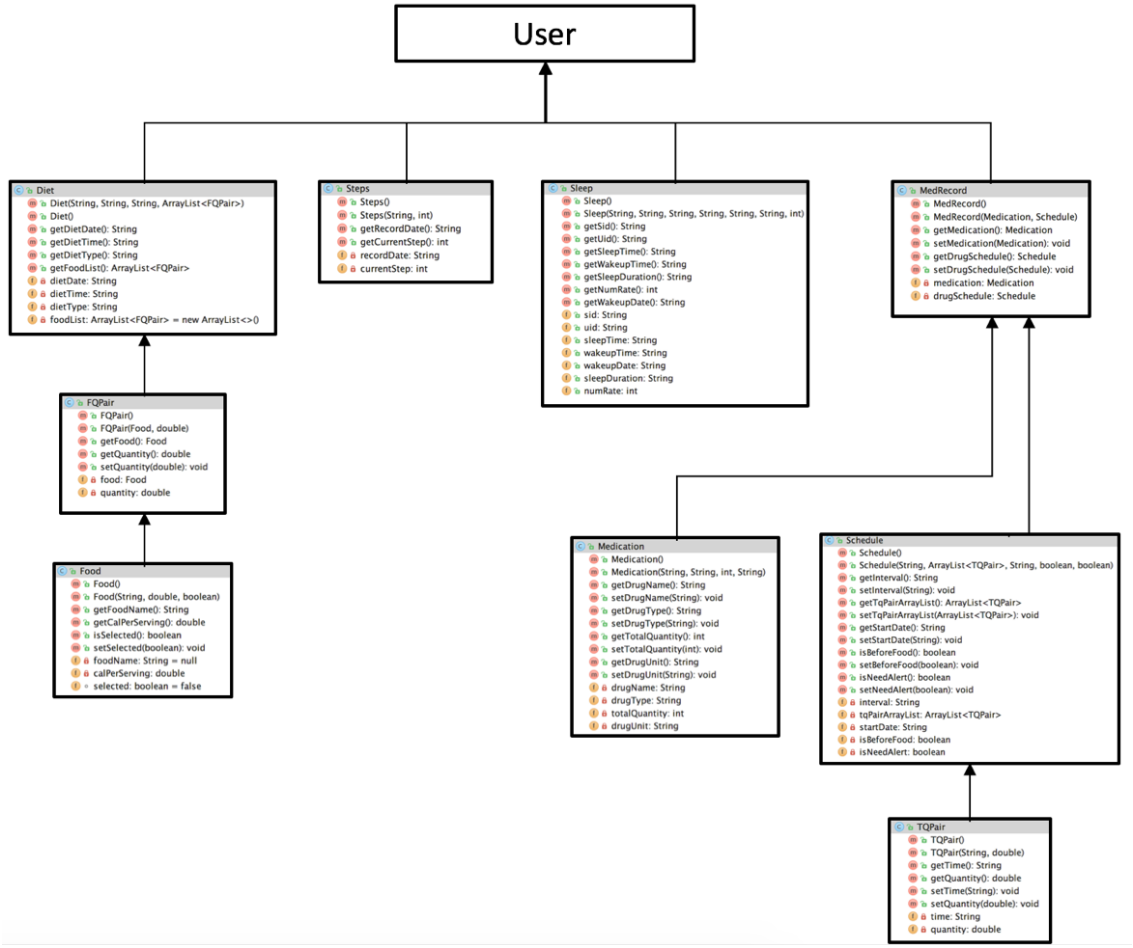
	https://github.com/DreamingCodeZH/MaterialRatingBar	
Firebase API (core, authentication, database)	com.google.firebase:firebase-core:10.2.0 com.google.firebase:firebase-auth:10.2.0 com.google.firebase:firebase-database:10.2.0 https://firebase.google.com/docs/android/setup	Providing connection support for Firebase authentication and database service.
Glide API	com.github.bumptech.glide:glide:3.7.0 https://github.com/bumptech/glide	An image loading and caching library for Android focused on smooth scrolling
butterknife	com.jakewharton:butterknife:7.0.1 https://github.com/JakeWharton/butterknife	Bind Android views and callbacks to fields and methods
Design support library	com.android.support:design:25.2.0	In this application, the design pattern follows the material design, including the color and view styles. All of the icons are collected from a free icon

	https://developer.android.com/training/material/design-library.html https://material.io/guidelines/style/color.html# https://icons8.com/icon/set	website which also provide material style icon.
Support library	com.android.support:support-v4:25.2.0 com.android.support:recyclerview-v7:25.2.0 com.android.support:cardview-v7:25.2.0 https://developer.android.com/topic/libraries/support-library/index.html	The android support library API is the center part that provide the UI styles and UI components.

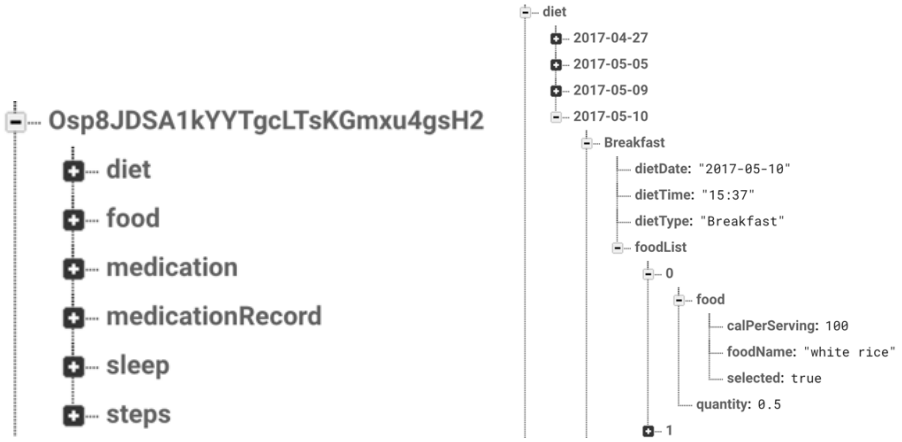
5.2 Database Interaction

As is introduced in the software design specification section, the database used in this application is Google Firebase database, a NoSQL document database. In this application, database interaction includes read, write, update and delete actions.

The database objects are defined as Java classes, stored in the application's "data" folder, including "User", "Diet", "Food", "FQPair", "MedRecord", "Medication", "Schedule", "Sleep", "Steps", and "TQPair". The detail of each Java classes is demonstrated in the figure below. As is shown in Figure 4-1, the "User" object holds "Sleep," "Steps," "Diet," and "MedRecord" four objects. Specifically, the "Diet" object holds an ArrayList of selected "FQPair" object and also other facts of a diet, while the "FQPair" object holds a pair of "Food" object and its quantity. The "Steps" object is much simpler, only includes the date and the steps. The "Sleep" object holds the date of sleep and other many sleep-related facts, the sleep duration is calculated by counting the difference between sleep time and awake time. Finally, the "MedRecord" object is the most complicated object here; it includes two objects, "Medication" and "Schedule." The "Medication" object holds the information of specific drug while the "Schedule" object holds the alert details specify by the user, it includes an array list of "TQPair" object which holds the drug's time and quantity.



In the Android program, each Java object is converted to JSON object and saved to Firebase database in .json format. Figure above shows the Firebase database structure when the user has tried all the functionalities provided by this program and Figure below shows the detailed structure of the “diet” entry.



To demonstrate four database interactions, the table below displays the database actions each Android functional pages performed and the targeted Java object for each action.

Database Actions/ Android Functional Pages	**Steps**	**Sleep**	**Diet**	**Medication**
Write	Steps object	Sleep object	Food object and Diet object	Medication object and MedRecord object
Read	Steps object	Sleep object	Food object and Diet object	Medication object and MedRecord object
Update	-	Sleep object	Food object and Diet object	Medication object and MedRecord object

Delete	-	Sleep object	Food object	Medication object and MedRecord object

5.3 Implementation Details

5.3.1 Implementation overview

As is mentioned in the software requirement specification part, this android application is implemented using Android Studio in Model-Presentation-Controller pattern. After we have learned about the database interaction (model) part, next, we will focus on the presentation (UI in action) part and the controller (the Android activities, fragments, and adapters) part. The UML diagram for the project is shown in Figure 5-1.

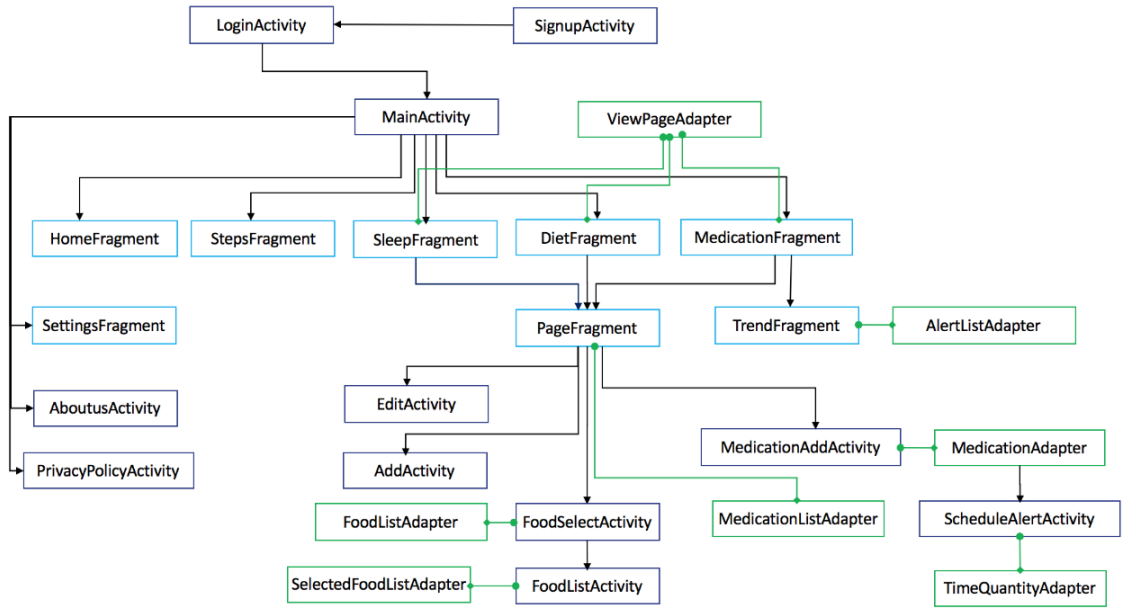


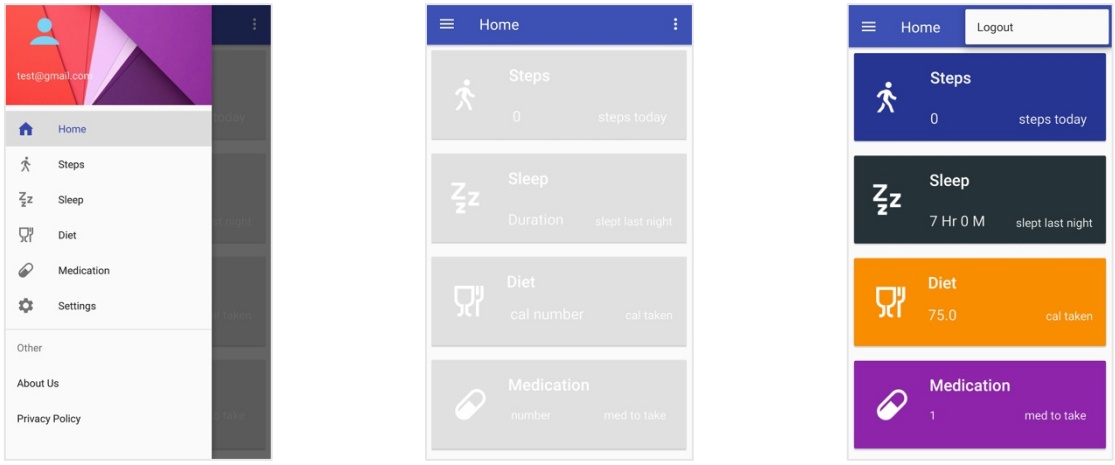
Figure 5-1 Architecture Diagram

5.3.2 Register and Login page

The register and login functionality are implemented by LoginActivity and SignupActivity, as above shows, SignupActivity will call the Login Activity when the user successfully registers an account, and the user should use the account to login in the LoginActivity. If the user authentication returns true, which means user login successfully, he will be direct to the MainActivity.

The user interface of the LoginActivity and SignupActivity is displayed in Figure above. The left figure displays an example of entering email and password to register a new user, and the icon in the right of the Password EditView is clickable, by click the icon, the user can hide or show its password. Also, both the Register and Login page support error detection. When user input strings that are not an email address, or input a weak password, password not much or some views are empty; the error message will show to the right of the EditView to remind the user about the error, see the middle figure for example.

5.3.3 Main Menu and Home page



After user successfully login to the system, the MainActivity will display a DrawerLayout navigation menu including all the functionalities this application can provides, see Figure above left.

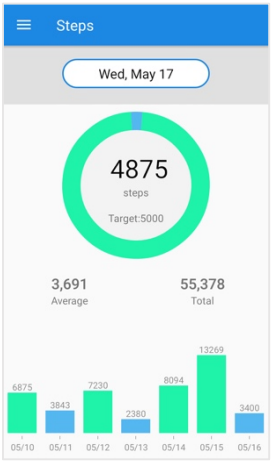
The home page is a summary of all the activities user has performed. It is displayed in a form of four CardView, if no data has been recorded in an activity, then the specific CardView will dispay a background color of grey as a reminder (see Figure above middle). However, if data is found, it will display each section's summary in a colorful form, see figure above.

The following code snippet demonstrates the HomeFragment displaying the steps CardView's data.

```
new Handler().post(new Runnable() {
    @Override
    public void run() {
        //Sleep

mRef.child(uid).child("steps").addListenerForSingleValueEvent(new
ValueEventListener() {
            @Override
            public void onDataChange(DataSnapshot dataSnapshot) {


                if(dataSnapshot.hasChild(selectedDate)){
                    //steps data exists
                    for (DataSnapshot sleepDataSnapshot :
```

```
dataSnapshot.getChildren()) {
                        Steps stepsObject =
sleepDataSnapshot.getValue(Steps.class);
//steps CardView  displays the current steps
stepsView.setText(String.valueOf(stepsObject.getCurrentStep()));

                    }//end valueListener
                }
                else{
                    //No steps data in record

cardStepsView.setBackgroundResource(R.color.md_grey_300);
                }

            }

            @Override
            public void onCancelled(DatabaseError databaseError) {
            }
        });//end fireBase listener
    }
});
```

5.3.4 Steps page

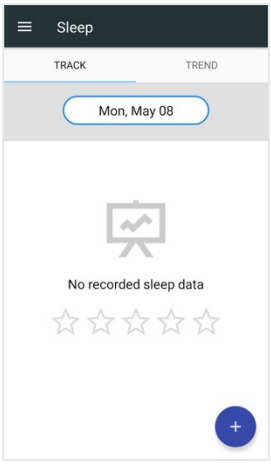


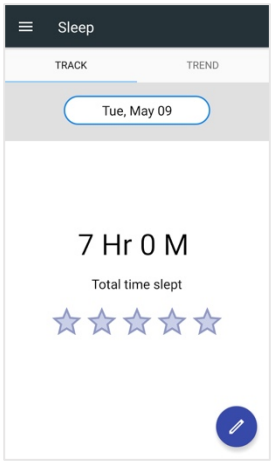
The steps page displays the current steps for today and also the last seven day's average, total steps and also specific steps for each day. The steps data is retrieved from sensor data provided by the Android API, if Android sensor detects changes, it will update the sensor's data. However, the steps counter from Android API can only provide the steps that since

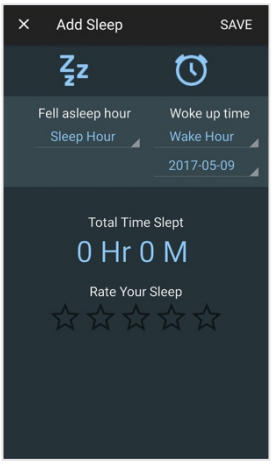
the device's last reboot, and is not what we want for a day's steps. Therefore, the following algorithm is used to calculate today's steps, see the following code snippet.

```
@Override
public void onSensorChanged(SensorEvent event) {

    //This is the algorithm to calculate the steps recorded by the Android
hardware
    //The current problem is the step increase too quick, not like it is true
    if (countSensorChange == 0){
        //First time to open the app
        since_boot = (int)(event.values[0]);
    }
    else{
        //The first time this app is opened
        currentStep = todayOffset + (int)(event.values[0]) - since_boot;

mRef.child(uid).child("steps").child(today).child("currentStep").setValue(curr
entStep);

    }
    int difference = goal - currentStep;
    if (difference > 0) {
        // goal not reached yet
        if (pg.getData().size() == 1) {
            pg.addPieSlice(sliceGoal);
        }
        sliceGoal.setValue(difference);
    } else {
        // goal reached
        pg.clearChart();
        pg.addPieSlice(sliceCurrent);
    }
    mStepsSinceReboot.setText(" "+currentStep);
    totalView.setText(formatter.format(total_start + currentStep));
    averageView.setText(formatter.format((total_start + currentStep) /
total_days));

    sliceCurrent.setValue(currentStep);
    pg.update();
    countSensorChange++;
}
```

5.3.5 Sleep page, Add Sleep page







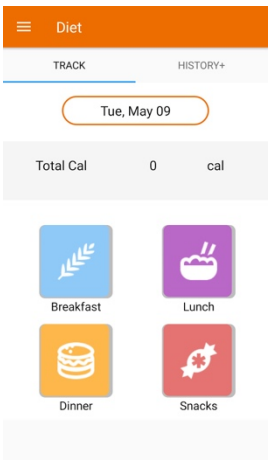
As is shown in figure above, the controller that is related to the Sleep page is SleepFragment, the controllers related to Add Sleep page are AddActivity and EditActivity.

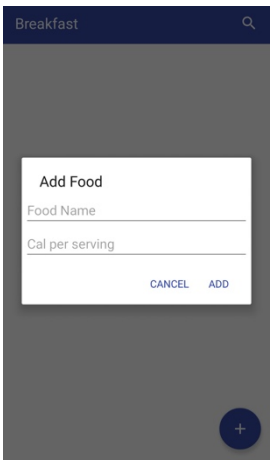
```
progress.show();
mRef.child(uid).child("sleep").addListenerForSingleValueEvent(new
ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {
        for (DataSnapshot sleepDataSnapshot : dataSnapshot.getChildren()) {


            sleepObject = sleepDataSnapshot.getValue(Sleep.class);

            if(sleepObject.getWakeupDate().equals(selectedDate)){

                currentSleepObject = sleepObject;
                sleepList.add(sleepObject);
                timeSlept = sleepObject.getSleepDuration();
                fab.setImageResource(R.drawable.ic_edit);
                tvDuration.setText(timeSlept);
                tvDuration.setBackgroundResource(R.color.white);

                tvStatus.setText("Total time slept");
                ratingBar.setRating(sleepObject.getNumRate());

                status = "edit";
                progress.dismiss();
            }//end if
        }//end valueListener
    }
    @Override
    public void onCancelled(DatabaseError databaseError) {
    }
});//end fireBase listener

mCountDownTimer = new CountDownTimer(1000000,1000) {
    @Override
```

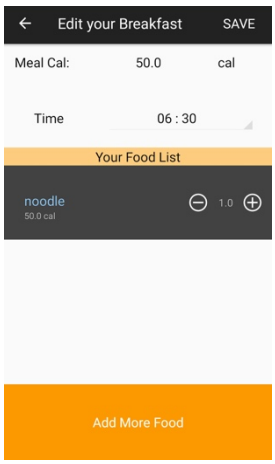
```
    public void onTick(long l) {
        if(sleepList.isEmpty()){
            progress.dismiss();
            fab.setImageResource(R.drawable.ic_plus);
            tvDuration.setText("");
            tvDuration.setBackgroundResource(R.drawable.ic_stat);
            tvStatus.setText("No recorded sleep data");
            status = "add";
            ratingBar.setRating(0);
        }

    }
    @Override
    public void onFinish() {}
};
mCountDownTimer.start();

fab.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {

        switch(status){

            case "add":

                //If add something
                Intent intentA = new Intent(getActivity(), AddActivity.class);
                Bundle extrasA = new Bundle();
                //Toast.makeText(getContext(), "add fab: " +
currentSleepObject.getWakeupDate() , Toast.LENGTH_LONG).show();
                extrasA.putString("EXTRA_TAG","Sleep");
                extrasA.putString("EXTRA_DATE",selectedDate);
                intentA.putExtras(extrasA);
                startActivity(intentA);

                break;

            case "edit":

                //If need to edit or delete record
                Intent intentE = new Intent(getActivity(), EditActivity.class);
                Bundle extrasE = new Bundle();

               //Toast.makeText(getContext(), "edit fab: " +
currentSleepObject.getWakeupDate(), Toast.LENGTH_LONG).show();
                extrasE.putString("EXTRA_TAG","Sleep");

extrasE.putString("EXTRA_DATE",currentSleepObject.getWakeupDate());

extrasE.putString("EXTRA_SLEEP_TIME",currentSleepObject.getSleepTime());

extrasE.putString("EXTRA_WAKE_TIME",currentSleepObject.getWakeupTime());

extrasE.putString("EXTRA_DURATION",currentSleepObject.getSleepDuration());
                extrasE.putString("EXTRA_RATE",
Integer.toString(currentSleepObject.getNumRate()));
                extrasE.putString("EXTRA_UID",uid);

                intentE.putExtras(extrasE);
                startActivity(intentE);
                break;
```

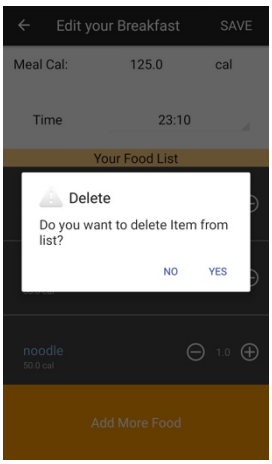
```
        }
    }
});
// on configuration changes (screen rotation) we want fragment member variables
to preserved
setRetainInstance(true);
```

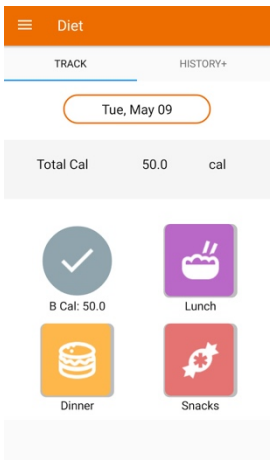
5.3.6 Diet page, Food Select page and diet edit page











To give a clear summary of relationship between the presentation and controller the table below list each presentation and the controller related to it and also their functions.

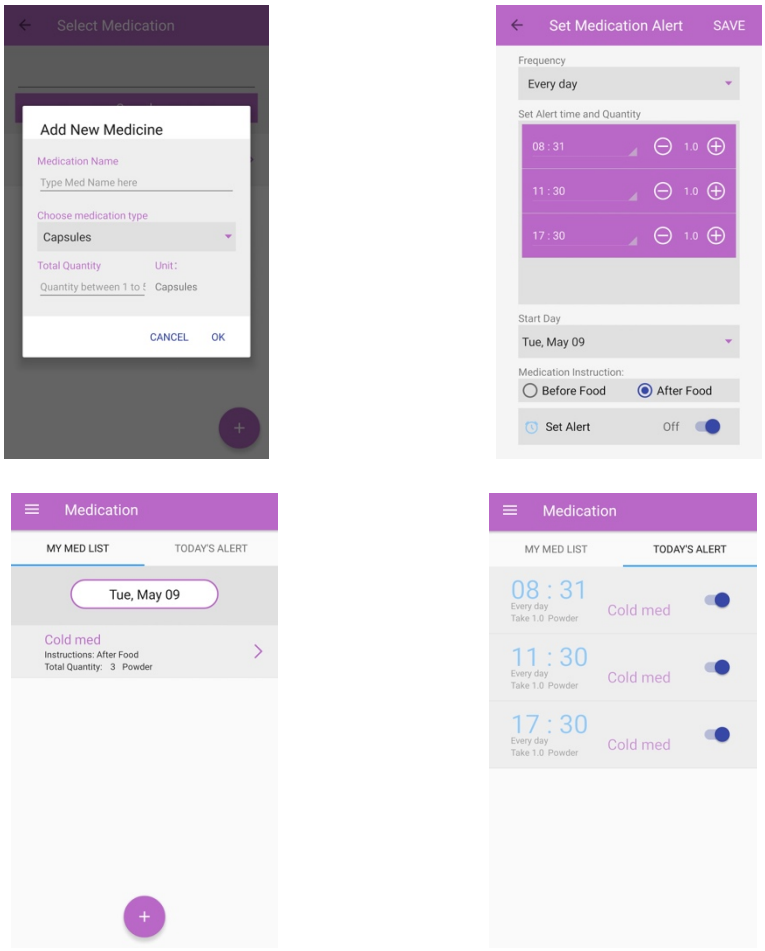
Presentation	Controller	Functions
Diet page	DietFragment, PageFragment	The controller will calculate the calories for each diet and also the total calories

		taken in the day, and display in each diet icon and also the total cal's textView.
Food Select Page	FoodSelectActivity, FoodListAdapter	This activity asks user to add new food to the list and select the added food, after the user select the food, he/she will be direct to the Diet Edit page.
Diet Edit Page	FoodListActivity, SelectedFoodListAdapter	In the diet edit page, the user can add the quantity of each food in the list and also add new food to the list.

```
//Start fetching data from database
//startDataFetch(selectedDate);
/**      Search by date time in database,
 *       If the record with the specific date time doesn't exist,
 *             display nothing
 *       If the time is not zero,
 *             then put the value in tvDuration and tvStatus text View
 **/

/* Attach a listener to read the data at our posts reference*/
progress.show();
mRef.child(uid).child("diet").child(selectedDate).addListenerForSingleValueEvent
(new ValueEventListener() {

    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {

        if(dataSnapshot.exists()){

            double totalCal = 0;
            for (DataSnapshot dietDataSnapshot : dataSnapshot.getChildren()) {

                dietObject = dietDataSnapshot.getValue(Diet.class);
                switch(dietObject.getDietType()) {

                    case "Breakfast" :
                        double calPerMealB = 0;
                        //Breakfast exists
                        breakfastBtn.setImageResource(R.drawable.ic_check);

breakfastBtn.setBackgroundResource(R.drawable.button_complete_back);
                        for(FQPair fqPair: dietObject.getFoodList()){
                            calPerMealB += fqPair.getQuantity()
*fqPair.getFood().getCalPerServing();
                        }
                        totalCal+=calPerMealB;
                        tvB.setText("B Cal: " + String.valueOf(calPerMealB));

                        break; // optional

                    case "Lunch" :
                        double calPerMealL = 0;
```

```
                        //Breakfast exists
                        lunchBtn.setImageResource(R.drawable.ic_check);

lunchBtn.setBackgroundResource(R.drawable.button_complete_back);
                        for(FQPair fqPair: dietObject.getFoodList()){
                            calPerMealL += fqPair.getQuantity()
*fqPair.getFood().getCalPerServing();
                        }
                        totalCal+=calPerMealL;
                        tvL.setText("L Cal: " + String.valueOf(calPerMealL));
                        break; // optional
                    case "Dinner" :
                        //Breakfast exists
                        double calPerMealD = 0;
                        dinnerBtn.setImageResource(R.drawable.ic_check);

dinnerBtn.setBackgroundResource(R.drawable.button_complete_back);
                        for(FQPair fqPair: dietObject.getFoodList()){
                            calPerMealD += fqPair.getQuantity()
*fqPair.getFood().getCalPerServing();
                        }
                        totalCal+=calPerMealD;
                        tvD.setText("D Cal: " + String.valueOf(calPerMealD));

                        break; // optional
                    case "Snacks" :
                        //Breakfast exists
                        double calPerMealS = 0;
                        snacksBtn.setImageResource(R.drawable.ic_check);

snacksBtn.setBackgroundResource(R.drawable.button_complete_back);
                        for(FQPair fqPair: dietObject.getFoodList()){
                            calPerMealS += fqPair.getQuantity()
*fqPair.getFood().getCalPerServing();
                        }
                        totalCal+=calPerMealS;
                        tvS.setText("S Cal: " + String.valueOf(calPerMealS));
                        break; // optional
                    // You can have any number of case statements.
                    default : // Optional
                }
                calPerDayTV.setText(String.valueOf(totalCal));

            }//end valueListener
            progress.dismiss();
        }
        progress.dismiss();


    }
    @Override
    public void onCancelled(DatabaseError databaseError) {
    }
});//end fireBase listener
```

5.3.7 Medication page, My Medication List page, Today's Alert page, Select Medication page and Set Medication Alert page



As one of the most complicated function in this application, there are five presentation component involved in the medication part, and nine controller associated with them, see Table below for detail association and the exact function of each page.

Presentation	Controller	Functions
Medication page	MedicationFragment	The holder of a tablayout including “My Medication List” page and the “Today’s alert” page.
My Medication List page	PageFragment, MedicationListAdapter	The left tab in the Medication page, “My Med List” display the medication record in a list view.

Select Medication page	MedicationAddActivity, MedicationAdapter	In the medication select pagem when user add a new mediaction by clicking the “add” button, and the medication will display in the medication list, when the user click the arrow in the right, he will be direct to the “Select medication alert” page.
Set Medication Alert page	ScheduleAlertActivity, TimeQuantityAdapter	In this page user can choose the detail information of the medication remider: the frequency and there are four time period he can add medication alert and he can also delete any one of the alert, and also the quantity of drug they choose to take at each time period.
Today’s Alert page	TrendFragment, AlertListAdapter	After the alert is set, it will be displayed in the “Today’s alert” tab, in the “Today’s Alert” page, user can turn on/turn off the alert.

```
public void generateList(){

    //Display all the food in the database to the Medication list

mRef.child(uid).child("medication").addListenerForSingleValueEvent(new
ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {

            for (DataSnapshot medDataSnapshot :
dataSnapshot.getChildren()) {


                if(medDataSnapshot.exists()){

                    Medication medObject =
medDataSnapshot.getValue(Medication.class);
                    medListAll.add(medObject);
                    listCounter++;
                }
                else{
                    progress.dismiss();
                    Toast.makeText(getApplicationContext(),"No medication
in history",Toast.LENGTH_LONG).show();
                }


            }//end valueListener
```

```
            progress.dismiss();
            adapter.notifyDataSetChanged(); // update adapter
        }
        @Override
        public void onCancelled(DatabaseError databaseError) { //do
nothing
        }
    }); //end fireBase listener

}//end Class generateList
```

6 Software Testing

6.1 Introduction and Motivation

Software testing is said to be the most beneficial part of software development. When including end user in every stage of testing, the software quality will approach the actual need of the user. The testing results will give a signal for the readiness of the application to be released to actual user and suggest future development direction. This document is intended to summarize and report the details of testing including testing environment, tools, types, and results. Also, it provides an evaluation of the testing results and a suggestion for further improvement.

6.2 Result Reporting

This section will describe the details of testing including testing environment, testing strategy and a summary of each testing methods.

6.2.1 Introduction

The testing is conducted for O Health, a health management application which is networking application with real-time data synchronization functionalities. The application includes four major functions: step counter, sleep recorder, diet recorder, and medication recorder and reminder. Therefore, the functions to be tested includes the database read/write delay, network speed, step counter accuracy for counting movement,

the ability of the application to work in different stress and the functionality correctness of each function.

6.2.2 Scope of Testing

a) In Scope

Functional Testing for the following modules are in Scope of Testing:

- Steps Counter
- Sleep Recorder
- Diet Recorder
- Medication Recorder and Reminder

b) Out of Scope

Security, regression, migration and disaster recovery testing are not performed due to the limited time and resource.

c) Items not tested

Currently, since we assume users to be able to connect to VPN and familiar with Android OS, therefore a survey of the VPN service's accessibility and user's familiarity to Android OS are not performed.

6.2.3 Acceptance Criteria

The acceptance criteria are defined as follows, according to Use Case in Software Requirement Specification Part:

User Registration and Login

I can register to the system by entering email and password.

I can access the system when system authentication succeeds.

I can edit and update my profile information.

I can verify my email address and change my password via email.

I can logout of the system.

User Health Profile Synchronization

I can login to difference devices (not simultaneously) and have my data synchronized with my specific account.

Steps Counting

I can view my steps by date and the past seven days' step history.

I can view my real-time steps.

Sleep Recording

I can view my sleep duration and sleep rate.

I can select a past date to view my history sleep duration and sleep rate.

I can add my sleep data for a selected date.

I can edit my sleep data if I want to change it and the page will keep my past record.

Diet Information Recording

I can select a diet type (breakfast, lunch, dinner, snacks) and edit my new diet by click the button.

I can search for a specific food by its name from a food list.

I can add a new food to the food list.

I can select foods from the food list and record my diet information including a diet time, quantity.

I can select a past date and view my past food record (if it exists).

I can view my calories for the whole day and also for each diet.

Medication Recorder and Alert

I can search for a specific drug by its name from a drug list.

I can add a new drug to the list.

I can select an existing drug and begin to edit the medication alert information including an interval of taking, schedule and dose time, start time, duration and instruction, whether the user wants an alter from the system.

I can view currently active medication list and my today's reminder.
Sending Health Profile I can send his medical profile to a designated email address.

6.2.4 Test Strategy

6.2.4.1 Test Approach

The following tests will be performed with the designed priority.

Sl. #	Type of Testing	Priority
1	Functionality: 1). Run Unit testing to confirm the application's output match with the designed expectation 2). Use Acceptance Criteria to compare the functionalities implementation with the designed use cases.	1
2	Performance: Run MONKEY service of the application and put the application under stress and measure its performance under stress	2
3	Usability/GUI: Design a GUI measure items and compare with the	4
4	Platform:	3

	Install application in different android platform and perform a performance test again	

6.2.4.2 Testing Environment and Tools

The table below demonstrates the testing environments

Testing Environment	OS: Android 5.0+
Testing Hardware Specification	MacBook Pro (CPU: 2.4 GHz, Intel Core i5; RAM: 8GB; Storage: 256 GB): running unit test, stress test and performance test code Android phone (Screen size: 5.5 inch; Screen Resolution: 1920 * 1080; Storage: 32GB; RAM: 3GB, LPDDR3; CPU: Helio P10 Processor): test device
Testing Network Setting	Wireless VPN-Connection: able to access Google service
Testing Database Setting	Google Firebase database
Testing Tools	Android Studio, Google Android MONKEY tool for stress test

6.2.5 Test Summary

6.2.5.1 Functionality Testing Summary

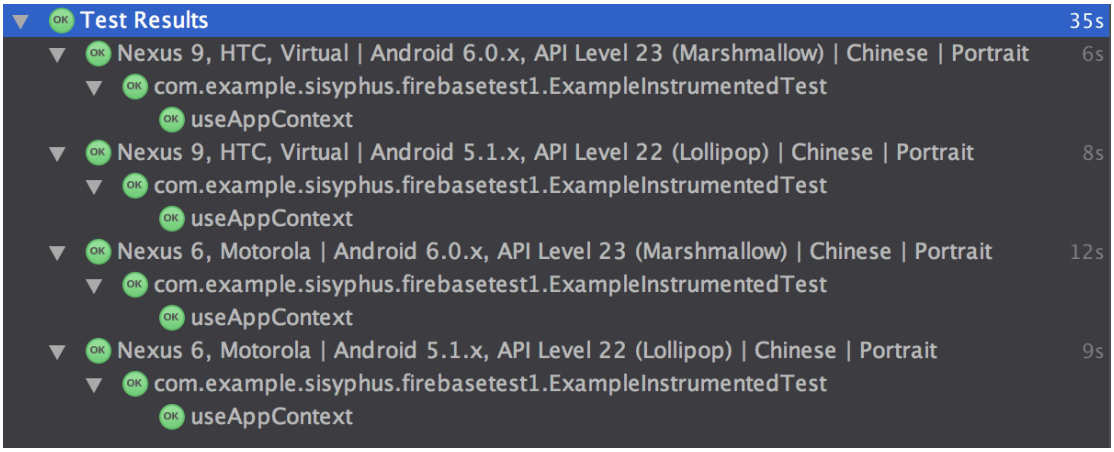
Aim: To confirm the application is complete and the output match with the the designed expectation.

Tool: Android Studio and JUnit

Strategy: (1) Test correctness of four functions: steps, sleep, diet and medication; (2) Compare the application with the Acceptance Criteria

Result:

(1) The application pass the JUnit test as displayed in the following figure.



(2) The criteria are matched with implementation except the last one "Sending Health Profile" as is demonstrated in the following table.

User Registration and Login ✓ I can register to the system by entering email and password. ✓I can access the system when system authentication succeeds. ✓I can edit and update my profile information. ✓I can verify my email address and change my password via email. ✓I can log out of the system.
User Health Profile Synchronization ✓I can login to difference devices (not simultaneously) and have my data synchronized with my specific account.
Steps Counting ✓I can view my steps by date and the past seven days' step history. ✓I can view my real-time steps.

Sleep Recording ✓I can view my sleep duration and sleep rate. ✓I can select a past date to view my history sleep duration and sleep rate. ✓I can add my sleep data for a selected date. ✓I can edit my sleep data if I want to change it and the page will keep my record.
Diet Information Recording ✓I can select a diet type (breakfast, lunch, dinner, snacks) and edit my new diet by click the button. ✓I can search for a specific food by its name from a food list. ✓I can add a new food to the food list. ✓I can select foods from the food list and record my diet information including a diet time, quantity. ✓I can select a past date and view my past food record (if it exists). ✓I can view my calories for the whole day and also for each diet.
Medication Recorder and Alert ✓I can search for a specific drug by its name from a drug list. ✓I can add a new drug to the list. ✓I can select an existing drug and begin to edit the medication alert information including an interval of taking, schedule and dose time, start time, duration and instruction, whether the user wants an alter from the system. ✓I can view currently active medication list and my today's reminder.
Sending Health Profile ✘I can send his medical profile to a designated email address.

6.2.5.2 Performance Testing Summary

Aim: Ensure the application can work under certain pressure

Tool: Android Studio, Monkey tools

Strategy: Run Android Monkey test

Result: the result of a stress test with 500 random touch event is that the application didn't exist while suffering from such a great amount of action and that it is reliable.

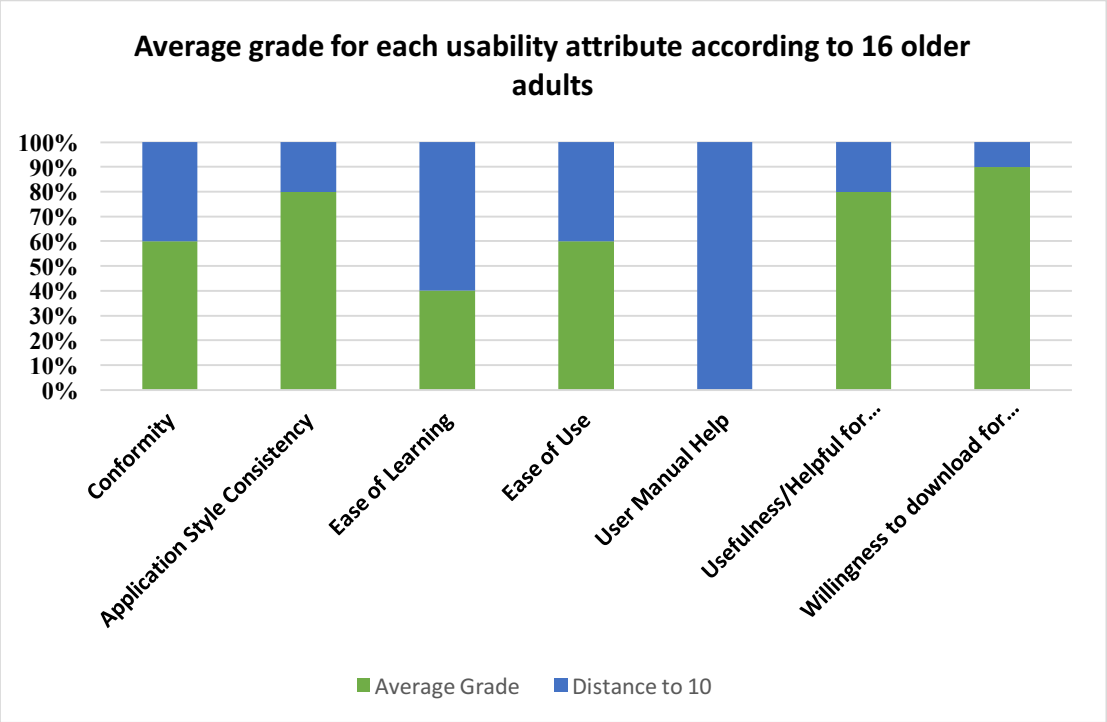
6.2.5.3 Usability Testing Summary

Aim: To test the application's effectiveness and user friendliness

Tool: Usability Matrix

Strategy: Use a matrix of criteria to measure the usability and ask 16 older adults to use this application and give Mark

Result: The results from the testing with older adults is demonstrated in the graph below. It shows that most of the older adults find the application easy to use with an average grade of 6, but still need be simplified, some of them suggested in the remark part that the application's Diet and Medication page should give user more alert rather than wait for the user to make their choice. With an average grade of 9, it shows that most of them will be happy to download this app. However, the grade for user manual help is 0, since it is not being taken into consideration and will be developed in the future.



6.2.5.4 Platform Testing Summary

Aim: To confirm the application can run on different Android platform

Tool: Android Studio and Firebase Test Lab

Strategy: Run Android Instrumented Unit Test and set

Result: The application can successfully run on different devices.

1.2 Conclusion and suggestion for future improvement

In conclusion, according to each testing results, this application has passed the unit test, various platform test, performance test and reach a level that is “useful.” However, future improvement is still required. As is mentioned above, a crucial function that allows user to send his/her health profile to a specific email address is still not implemented, and the user manual is still not designed. Also, the ease of use of this application is still required improvement to become simpler and easier to use.

7 Discussion

This project is an initial attempt to design and implement a mobile application targeting older adults' health self-management, with its high much between the software requirements and design specifications with the actual implementation results; it is proved that its project is with high feasibility. Also, the software performance test and the usability acceptance test reveal a satisfactory performance of the software and user feedbacks.

8 Future Work/Conclusion

Due to time and limit of resource, some design has still not yet implemented (because of their low priority compared to others), and they should be left to the future implementation. Also, the usability test has collected many feedbacks that should be added to the future improvements.

Firstly, it is suggested that adding a user group called caretaker and separate their functionality from the original user group the older adults. The caretaker account should be able to associate with one or many older adult's accounts and record their diet, drugs need to take. On the other hand, the older adult account should be able to count user's steps, and allow the user to view the record of steps, sleep and diet page and also see the instruction of care-takers on medication page. Also, when the medication reminder is on, he can close the reminder and meanwhile give feedback about this medication's taking.

Secondly, for the steps and sleep recorder, some user suggests that the application should not ask user to record sleep manually, and also improve the accuracy of the step counter. A good choice is to connect to a smart band and let the band send data

to this application and display the data to the user. Although this functionality is not implemented now since the inadequate open-source API for some smart band, it is possible to be implemented when the more developer-friendly smart band come to the market.

Finally, for the Diet recorder, currently, the user should enter food's name and calories manually to add a new food. In this way, the calories counting is not only inaccurate but also labor-taking. Therefore, in the future, it should connect to an open-source food database and allow the user to search food by name in that database or more wisely, the user can use the QR code in the food's packet and get all the information of the food.

9 References

1. Kumar, S., Nilsen, W. J., Abernethy, A., Atienza, A., Patrick, K., Pavel, M., ... & Hedeker, D. 2013. Mobile health technology evaluation: the mHealth evidence workshop. *American journal of preventive medicine*, *45*(2), 228-236.
2. Wynne Wang. 2016. The Sliver Age: China's Aging Population. http://knowledge.ckgsb.edu.cn/2016/10/17/demographics/silver-age-chinas-aging-population/
3. Irvine, A. B., Gelatt, V. A., Seeley, J. R., Macfarlane, P., & Gau, J. M. (2013). Web-based intervention to promote physical activity by sedentary older adults: randomized controlled trial. *Journal of medical Internet research*, *15*(2), e19.
4. World Health Organization. (2015). China country assessment report on ageing and health.

5. Demiris, G., Thompson, H. J., Reeder, B., Wilamowska, K., & Zaslavsky, O. (2013). Using informatics to capture older adults' wellness. International journal of medical informatics, 82(11), e232-e241.
6. Heart, T., & Kalderon, E. (2013). Older adults: are they ready to adopt health-related ICT? *International journal of medical informatics*, *82*(11), e209-e231.
7. Chen, K., Chan, A. H., & Tsang, S. N. (2013). Usage of mobile phones amongst elderly people in Hong Kong.
8. Deng, Z., Mo, X., & Liu, S. (2014). Comparison of the middle-aged and older users' adoption of mobile health services in China. *International journal of medical informatics*, *83*(3), 210-224.
9. Xue, L., Yen, C. C., Chang, L., Chan, H. C., Tai, B. C., Tan, S. B., ... & Choolani, M. (2012). An exploratory study of ageing women's perception on access to health informatics via a mobile phone-based intervention. *International journal of medical informatics*, *81*(9), 637-648.
10. Grindrod, K. A., Li, M., & Gates, A. (2014). Evaluating user perceptions of mobile medication management applications with older adults: a usability study. JMIR mHealth and uHealth, 2(1), e11.
11. Joe, J., & Demiris, G. (2013). Older adults and mobile phones for health: A review. Journal of biomedical informatics, 46(5), 947-954.

Appendix A: Android Monkey Test Result Code

:Monkey: seed=1493986796272 count=500

:AllowPackage: com.example.sisyphus.firebasetest1

:IncludeCategory: android.intent.category.LAUNCHER

:IncludeCategory: android.intent.category.MONKEY

// Event percentages:

//   0: 15.0%

//   1: 10.0%

//   2: 2.0%

//   3: 15.0%

//   4: -0.0%

//   5: 25.0%

//   6: 15.0%

//   7: 2.0%

//   8: 2.0%

//   9: 1.0%

//   10: 13.0%

:Switch:
#Intent;action=android.intent.action.MAIN;category=android.intent.category.LAUNCHER;launchFlags=0x10200000;component=com.example.sisyphus.firebasetest1/.activity.LoginActivity;end

    // Allowing start of Intent { act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] cmp=com.example.sisyphus.firebasetest1/.activity.LoginActivity } in package com.example.sisyphus.firebasetest1

:Sending Touch (ACTION_DOWN): 0:(810.0,541.0)

:Sending Touch (ACTION_UP): 0:(834.4437,422.48016)

:Sending Trackball (ACTION_MOVE): 0:(-2.0,2.0)

:Sending Touch (ACTION_DOWN): 0:(430.0,1015.0)

:Sending Touch (ACTION_UP): 0:(430.12268,995.452)

:Sending Touch (ACTION_DOWN): 0:(592.0,1567.0)

:Sending Touch (ACTION_UP): 0:(582.06647,1570.8824)

:Sending Trackball (ACTION_MOVE): 0:(3.0,3.0)

:Sending Touch (ACTION_DOWN): 0:(76.0,1545.0)

:Sending Touch (ACTION_UP): 0:(90.58824,1532.5339)

:Sending Trackball (ACTION_MOVE): 0:(2.0,-2.0)

:Sending Touch (ACTION_DOWN): 0:(950.0,931.0)

:Sending Touch (ACTION_UP): 0:(952.69836,946.44507)

:Sending Touch (ACTION_DOWN): 0:(1074.0,757.0)

:Sending Touch (ACTION_UP): 0:(1054.0251,748.2423)

:Sending Trackball (ACTION_MOVE): 0:(2.0,3.0)

:Sending Touch (ACTION_DOWN): 0:(753.0,80.0)

:Sending Touch (ACTION_UP): 0:(741.0721,69.933754)

:Sending Flip keyboardOpen=false

Got IOException performing flipjava.io.IOException: write failed: EINVAL (Invalid argument)

// Injection Failed

//[calendar_time:2017-05-01 23:36:13.047 system_uptime:1186814]

// Sending event #100

:Sending Touch (ACTION_DOWN): 0:(864.0,1801.0)

:Sending Touch (ACTION_UP): 0:(877.90955,1814.2079)

:Sending Trackball (ACTION_MOVE): 0:(-2.0,-1.0)

:Sending Touch (ACTION_DOWN): 0:(82.0,764.0)

:Sending Touch (ACTION_UP): 0:(84.07447,767.1773)

:Sending Touch (ACTION_DOWN): 0:(482.0,1626.0)

:Sending Touch (ACTION_UP): 0:(481.6873,1628.0245)

:Sending Trackball (ACTION_MOVE): 0:(2.0,1.0)

:Sending Touch (ACTION_DOWN): 0:(656.0,1599.0)

:Sending Touch (ACTION_UP): 0:(669.1314,1532.7084)

:Sending Touch (ACTION_DOWN): 0:(855.0,784.0)

:Sending Touch (ACTION_UP): 0:(859.6757,804.7579)

:Sending Trackball (ACTION_MOVE): 0:(-2.0,-1.0)

:Sending Touch (ACTION_DOWN): 0:(366.0,1252.0)

:Sending Touch (ACTION_UP): 0:(433.72412,1321.2706)

:Sending Touch (ACTION_DOWN): 0:(116.0,1493.0)

:Sending Touch (ACTION_UP): 0:(116.61606,1493.4775)

:Sending Touch (ACTION_DOWN): 0:(823.0,362.0)

:Sending Touch (ACTION_UP): 0:(821.69116,360.0197)

:Sending Touch (ACTION_DOWN): 0:(688.0,917.0)

:Sending Touch (ACTION_UP): 0:(682.9219,924.8812)

:Sending Touch (ACTION_DOWN): 0:(117.0,299.0)

//[calendar_time:2017-05-01 23:36:13.615 system_uptime:1187339]

// Sending event #200

:Sending Touch (ACTION_UP): 0:(68.69516,226.28812)

:Sending Touch (ACTION_DOWN): 0:(917.0,1792.0)

:Sending Touch (ACTION_UP): 0:(900.8629,1776.609)

:Sending Trackball (ACTION_MOVE): 0:(-3.0,-3.0)

:Sending Touch (ACTION_DOWN): 0:(744.0,261.0)

:Sending Touch (ACTION_UP): 0:(751.5805,261.59927)

:Sending Touch (ACTION_DOWN): 0:(419.0,1881.0)

:Sending Touch (ACTION_UP): 0:(442.14984,1879.4745)

:Sending Trackball (ACTION_MOVE): 0:(4.0,-4.0)

:Sending Touch (ACTION_DOWN): 0:(56.0,676.0)

:Sending Touch (ACTION_UP): 0:(13.647469,701.2936)

:Sending Touch (ACTION_DOWN): 0:(360.0,1191.0)

:Sending Touch (ACTION_UP): 0:(342.85846,1166.9926)

:Sending Trackball (ACTION_MOVE): 0:(-2.0,-4.0)

:Sending Trackball (ACTION_MOVE): 0:(-3.0,-2.0)

:Sending Trackball (ACTION_MOVE): 0:(-4.0,-3.0)

//[calendar_time:2017-05-01 23:36:13.765 system_uptime:1187489]

// Sending event #300

:Sending Touch (ACTION_DOWN): 0:(424.0,1243.0)

:Sending Touch (ACTION_UP): 0:(427.11273,1253.7635)

:Sending Trackball (ACTION_MOVE): 0:(0.0,0.0)

:Sending Flip keyboardOpen=true

Got IOException performing flipjava.io.IOException: write failed: EINVAL (Invalid argument)

// Injection Failed

:Sending Touch (ACTION_DOWN): 0:(920.0,1740.0)

:Sending Touch (ACTION_UP): 0:(906.6936,1739.257)

:Sending Touch (ACTION_DOWN): 0:(409.0,1705.0)

:Sending Touch (ACTION_UP): 0:(416.63132,1697.2347)

:Sending Touch (ACTION_DOWN): 0:(774.0,187.0)

:Sending Touch (ACTION_UP): 0:(769.25305,167.65448)

:Switch:
#Intent;action=android.intent.action.MAIN;category=android.intent.category.LAUNCHER;launchFl
ags=0x10200000;component=com.example.sisyphus.firebasetest1/.activity.LoginActivity;end

    // Allowing start of Intent { act=android.intent.action.MAIN
cat=[android.intent.category.LAUNCHER]
cmp=com.example.sisyphus.firebasetest1/.activity.LoginActivity } in package
com.example.sisyphus.firebasetest1

:Switch:
#Intent;action=android.intent.action.MAIN;category=android.intent.category.LAUNCHER;launchFl
ags=0x10200000;component=com.example.sisyphus.firebasetest1/.activity.LoginActivity;end

    // Allowing start of Intent { act=android.intent.action.MAIN
cat=[android.intent.category.LAUNCHER]
cmp=com.example.sisyphus.firebasetest1/.activity.LoginActivity } in package
com.example.sisyphus.firebasetest1

    // Rejecting start of Intent { act=android.intent.action.MAIN
cat=[android.intent.category.LAUNCHER]
cmp=android/com.android.internal.app.MzResolverActivity } in package android

:Sending Trackball (ACTION_MOVE): 0:(0.0,-1.0)

:Sending Trackball (ACTION_MOVE): 0:(-4.0,2.0)

:Sending Touch (ACTION_DOWN): 0:(813.0,865.0)

:Sending Touch (ACTION_UP): 0:(819.0796,865.0271)

:Sending Trackball (ACTION_MOVE): 0:(-5.0,-4.0)

:Sending Touch (ACTION_DOWN): 0:(931.0,682.0)

:Sending Touch (ACTION_UP): 0:(932.3905,690.52264)

:Sending Touch (ACTION_DOWN): 0:(577.0,1729.0)

:Sending Touch (ACTION_UP): 0:(570.18463,1719.4592)

:Sending Trackball (ACTION_MOVE): 0:(-1.0,-1.0)

    //[calendar_time:2017-05-01 23:36:14.194 system_uptime:1187918]

    // Sending event #400

:Sending Trackball (ACTION_MOVE): 0:(3.0,3.0)

:Sending Trackball (ACTION_UP): 0:(0.0,0.0)

:Sending Flip keyboardOpen=false

Got IOException performing flipjava.io.IOException: write failed: EINVAL (Invalid argument)

// Injection Failed

:Sending Trackball (ACTION_MOVE): 0:(4.0,-2.0)

:Sending Touch (ACTION_DOWN): 0:(1078.0,1342.0)

:Sending Touch (ACTION_UP): 0:(1080.0,1253.1116)

:Sending Touch (ACTION_DOWN): 0:(281.0,524.0)

:Sending Touch (ACTION_UP): 0:(373.44766,453.86746)

:Sending Touch (ACTION_DOWN): 0:(707.0,165.0)

:Sending Touch (ACTION_UP): 0:(707.3202,189.72226)

:Sending Touch (ACTION_DOWN): 0:(862.0,308.0)

:Sending Touch (ACTION_UP): 0:(852.7386,303.0145)

:Sending Trackball (ACTION_MOVE): 0:(2.0,-1.0)

Events injected: 500

:Sending rotation degree=0, persist=false

// Rejecting start of Intent
{ cmp=com.android.systemui/com.flyme.systemui.recents.RecentsEmptyActivity } in package com.android.systemui

:Dropped: keys=0 pointers=0 trackballs=0 flips=3 rotations=0

Network stats: elapsed time=2867ms (0ms mobile, 0ms wifi, 2867ms not connected)

// Monkey finished

Appendix B: Usability Testing Matrix

#	Attribute	Grade	Remark
1	Conformity	0-10	10 is the best

2	Application Style Consistency	0-10	
3	Ease of Learning	0-10	
4	Ease of Use	0-10	
5	User Manual Help	0-10	
6	Usefulness/Helpful for Health Care Management	0-10	
7	Willingness to download for free	0-10	