# ANDROID-BASED DIET CONSULTANT

**TNEK SHIIN WEI**

**UNIVERSITI TUNKU ABDUL RAHMAN**

# ANDROID-BASED DIET CONSULTANT

**TNEK SHIIN WEI**

**A project report submitted in partial fulfilment of the requirements for the award of Bachelor of Science (Honours) Software Engineering**

**Lee Kong Chian Faculty of Engineering and Science**
**Universiti Tunku Abdul Rahman**

**SEPTEMBER 2023**

**DECLARATION**

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

| | | |
|---|---|---|
| Signature | : | Shiin |
| Name | : | Tnek Shiin Wei |
| ID No. | : | 20UEB05626 |
| Date | : | 4/10/2023 |

**APPROVAL FOR SUBMISSION**

I certify that this project report entitled "**ANDROID-BASED DIET CONSULTING APPLICATION**" was prepared by **TNEK SHIIN WEI** has met the required standard for submission in partial fulfilment of the requirement for the award of Bachelor of Science (Honours) Software Engineering at Universiti Tunku Abdul Rahman

Approved by,

Signature : _____

Supervisor : Beh Hooi Ching

Date : 4th October 2023

Signature : _____

Co-Supervisor : _____

Date : _____

# ACKNOWLEDGEMENTS

I would like to express my gratitude to my supervisor, Ms. Beh Hooi Ching, for her invaluable advice and guidance throughout the development of the project. Her support and experience have been invaluable to me in completing this project.

Additionally, I would like to extend my gratitude to my loving family members and friends who have provided me with support and encouragement throughout my degree studies.

Lastly, I would also like to thank the testers who agreed to participate in the project's user acceptance testing. The feedback provided has been valuable and informative, contributing to the project's improvements.

## ABSTRACT

In a world where health risks related to diet are of growing concern, the Android-based Diet Consultant application emerges as a proactive solution. This application is meticulously crafted to simplify and promote healthy dietary practices and lifestyle choices by comparing the existing implementation in the market as explored in Chapter 2 during the Literature Review. Serving as a virtual dietitian, it empowers users to effortlessly generate personalized diet plans, as detailed in Chapter 4, eliminating the need for traditional in-person consultations. This not only saves valuable time but also allows users to create tailored dietary regimens based on their individual data with unparalleled ease, a process outlined in Chapter 4 as part of the Project Specification. Beyond diet planning, as discussed in Chapter 5 during the Design phase, the application actively monitors and tracks user progress, providing insights and feedback for a holistic health journey. Moreover, the inclusion of an AI diet chatbot, implemented in Chapter 6 during the Implementation phase, further enhances the user experience by offering practical solutions to contemporary health inquiries. Harnessing the power of artificial intelligence, as detailed in Chapter 6, the Android-based Diet Consultant application empowers users to seize control of their health and contribute to a healthier, more informed future. As we delve deeper into the chapters, we will explore the meticulous methodologies and work plan, as presented in Chapter 3, the detailed system architecture design in Chapter 5, and the comprehensive testing and evaluation process in Chapter 7. Finally, in Chapter 8, we conclude by reflecting on the achievement of objectives, acknowledging limitations, and providing recommendations for future work.

# TABLE OF CONTENTS

**TABLE OF CONTENTS** **i**
**LIST OF TABLES** **v**
**LIST OF FIGURES** **vi**
**LIST OF SYMBOLS / ABBREVIATIONS** **xi**
**LIST OF APPENDICES** **xii**

**CHAPTER**

**1 INTRODUCTION** **1**
1.1 Introduction 1
1.2 Problem Statement 2
1.2.1 Imbalanced dietary 2
1.2.2 Increasing indiscipline lifestyle 3
1.2.3 Increasing mental health issue 3
1.3 Aim and Objectives 4
1.4 Proposed Solution 5
1.5 Proposed Approach 6
1.5.1 Research Approach 6
1.5.2 Development Approach 6
1.6 Project Scope 8
**2 LITERATURE REVIEW** **10**
2.1 Introduction 10
2.2 Differences between dietitian and nutritionist 12
2.3 Similar Systems Review 12
2.3.1 Existing Application Comparative Analysis 13
2.4 Body Mass Index (BMI) 20
2.5 Basal Metabolic Rate (BMR) 22
2.6 Percentage of Fats 23
2.7 Muscle Mass 25

| | | | |
|---|---|---|---|
| | 2.8 | Rule-Based System Technique (RBS) | 26 |
| | 2.9 | Decision Trees | 27 |
| | 2.10 | Support Vector Machines | 29 |
| | 2.11 | Summary | 30 |
| **3** | **METHODOLOGY AND WORK PLAN** | | **31** |
| | 3.1 | Introduction | 31 |
| | 3.2 | Incremental and Iterative Development Methodology | 31 |
| | | 3.2.1 Planning | 33 |
| | | 3.2.2 Analysis and Design | 35 |
| | | 3.2.3 Design, Development, and Testing | 36 |
| | | 3.2.4 Closing | 40 |
| | 3.3 | Tools Used in This Project | 41 |
| | | 3.3.1 Axure Rp | 41 |
| | | 3.3.2 Visual Studio Code and Android Studio | 41 |
| | | 3.3.3 Flutter | 41 |
| | | 3.3.4 Drawio | **Error! Bookmark not defined.** |
| | | 3.3.5 Enterprise Architect | 42 |
| | | 3.3.6 Firebase | 42 |
| | 3.4 | Work Breakdown Structure (WBS) | 43 |
| | 3.5 | Gantt Chart | 47 |
| | | 3.5.1 Overview of the Project Timeline | 47 |
| | | 3.5.2 Planning Phase Timeline | 48 |
| | | 3.5.3 Analysis and Design Phase Timeline, and The Transition between Project I and II | 48 |
| | | 3.5.4 Design, Development, and Testing Iteration 1 Timeline | 49 |
| | | 3.5.5 Design, Development, and Testing Iteration 2 Timeline | 49 |
| | | 3.5.6 Design, Development, and Testing Iteration 3 Timeline | 49 |
| | | 3.5.7 Design, Development and Testing Iteration 4 Timeline | 50 |

| | | | |
|---|---|---|---|
| | | 3.5.8 Design, Development, and Testing Final Iteration Timeline | 50 |
| | | 3.5.9 Closing Phase Timeline | 50 |
| | 3.6 | Summary | 51 |
| **4** | **PROJECT SPECIFICATION** | | **52** |
| | 4.1 | Introduction | 52 |
| | 4.2 | Facts Finding | 52 |
| | | 4.2.1 Demographics of respondents | 52 |
| | | 4.2.2 General question about user's current diet | 53 |
| | | 4.2.3 Diet consultant application related questions among users | 56 |
| | 4.3 | Requirement Specification | 64 |
| | | 4.3.1 Functional Requirement | 65 |
| | | 4.3.2 Non-Functional Requirement | 66 |
| | 4.4 | System Use Case | 67 |
| | | 4.4.1 Use Case Diagram | 67 |
| | | 4.4.2 Use Case Description | 68 |
| | 4.5 | Interface Flow Diagram | 77 |
| | 4.6 | Prototype | 78 |
| | | 4.6.1 Mobile Application Interface Design | 78 |
| **5** | **DESIGN** | | **86** |
| | 5.1 | Introduction | 86 |
| | 5.2 | System Architecture Design | 86 |
| | 5.3 | Data Model Design | 87 |
| | | 5.3.1 System Database Design | 88 |
| | | 5.3.2 Data Dictionary | 89 |
| | | 5.3.3 Activity Diagram | 91 |
| | 5.4 | User Interface Designs | 99 |
| **6** | **IMPLEMENTATION** | | **104** |
| | 6.1 | Introduction | 104 |
| | 6.2 | Flutter Libraries | 104 |
| | 6.3 | Firebase | 106 |
| | | 6.3.1 Implementation of Firebase Authentication | 106 |
| | | 6.3.2 Sign in | 107 |

6.3.3 Sign Up 108
6.3.4 Implementation of Firebase Firestore Database 110
6.4 Implementation of OpenAI API 116
6.4.1 Generate diet plan 117
6.4.2 Chatbot 118
6.5 Summary 118
**7 TESTING AND EVALUATION 119**
7.1 Introduction 119
7.2 Traceability between Use Cases, Functional Requirements and Test Cases 119
7.2.1 Use Case Table 120
7.2.2 Functional Requirement Table 120
7.3 Unit Testing 121
7.3.1 Unit Test Results 125
7.4 Integration Testing 125
7.4.1 Test Cases Table of Integration Testing 125
7.4.2 Integration Testing Result 127
7.5 User Acceptance Test 127
7.5.1 User Acceptance Test Plan 127
7.5.2 User Acceptance Test Cases 128
7.5.3 User Acceptance Test Result 129
7.5.4 User Acceptance Test Result Summary 129
7.6 Summary 130
**8 CONCLUSION AND RECOMMENDATION 132**
8.1 Introduction 132
8.2 Objective Achievement 133
8.3 Limitations 134
8.4 Recommendation for Future Work 136
**REFERENCES 138**
**APPENDICES 141**

# LIST OF TABLES

Table 2.1: Features comparison among the existing similar application 18

Table 2.2: BMI (Anggraeni, n.d.) 20

Table 2.3: BMR Activity and Factor 22

Table 2.4: Healthy body fat percentage for men and women (Forbes, n.d.). 24

Table 2.5: Muscle mass percentage (Kirsten Nunez, 2019). 25

Table 5.1: Data dictionary for users table. 89

Table 5.2 : Data dictionary for dietPlan table. 89

Table 5.3 : Data dictionary for feedback table. 90

Table 5.4 : Data dictionary for userProgress table. 90

Table 7.1 Use Cases 120

Table 7.2: Functional Requirements 120

Table 7.3: Unit testing test cases 121

Table 7.4: Integration Testing test cases 125

Table 7.5 User Acceptance Test Cases 128

Table 7.6: UAT participants age group 130

Table 7.7: UAT feedback result summary 130

Table 8.1: Recommendations 136

# LIST OF FIGURES

Figure 1.1.1 : System Architecture Overview 5

Figure 1.2 :Sample pie chart collected in Google Form. 6

Figure 2.1: MyFitnessPal 13

Figure 2.2: MyNetDiary 14

Figure 2.3: Lifesum 15

Figure 2.4: My Diet Coach (Anon, 2022). 16

Figure 2.5: Lost It! 17

Figure 2.6: Body Mass Index (BMI) Chart for Adults (Wittwer, 2018) 21

Figure 2.7 Basic architecture of rule-based system (Salah Gad Foda, 2012). 26

Figure 2.8 Decision Tree Element Illustration (Kosarenko, 2021) 27

Figure 2.9: Illustration of SVM's supervised learning algorithms (Raj, 2022). 29

Figure 3.1: SDLC's iterative and incremental model illustration (Anon, 2019). 33

Figure 3.2: Overview of the Project Timeline. 47

Figure 3.3: Planning Phase Timeline. 48

Figure 3.4: Analysis and Design Phase Timeline including the Transition between Project I and II. 48

Figure 3.5: Design, Development and Testing Iteration 1 Timeline. 49

Figure 3.6: Design, Development and Testing Iteration 2 Timeline. 49

Figure 3.7: Design, Development and Testing Iteration 3 Timeline. 49

Figure 3.8: Design, Development and Testing Iteration 4 Timeline. 50

Figure 3.9: Design, Development and Testing Final Iteration Timeline. 50

Figure 3.10: Closing Phase Timeline. 50

Figure 4.1: Age distribution among the 40 respondents. 53

Figure 4.2: Gender of the 40 respondents 53

Figure 4.3: Description of 40 respondents on their current diet. 53

Figure 4.4: Biggest challenge of 40 respondents in following a healthy diet. 54

Figure 4.5: Frequency of 40 respondents go for a diet consultation session physically. 55

Figure 4.6: Level of strictness of the 38 respondents follow to the diet plan provided by their dietician. 55

Figure 4.7: Positive changes of the 40 respondents in their diet or health after consulting a dietician. 56

Figure 4.8: Responses of the 40 respondents on whether they ever used a diet consulting application before. 57

Figure 4.9: Expected features in diet consulting application from 40 respondents. 57

Figure 4.10: Primary goals of 40 respondents seeking from a diet consulting application. 58

Figure 4.11: Level of importance for 40 respondents for a diet consulting application to provide a personalized meal plan. 58

Figure 4.12: Willingness of 40 respondents to pay for a diet consulting application. 59

Figure 4.13: Importance of credibility of the dietitian or nutritionist who created the application's meal plans and recommendations. 60

Figure 4.14: Respondent's likeliness of recommendation of diet consulting application to their friends or family members. 60

Figure 4.15: Importance of a diet consulting application to provide recipes. 61

Figure 4.16: Satisfaction with currently use diet consulting application. 61

Figure 4.17: Importance of diet consulting application to have a community aspect. 62

Figure 4.18: Likeliness of continuous usage of diet consulting applications in the future. 62

Figure 4.19: Likeliness of 40 respondents receiving updates or reminders from a diet consulting application. 63

Figure 4.20: Availability of smartwatch among 40 respondents. 64

Figure 4.21: Use case diagram for the proposed application 67

Figure 4.22: Interface flow diagram of the proposed system 77

Figure 4.23: Welcome Screen. 78

Figure 4.24: Sign Up and Login Screen. 79

Figure 4.25: Home Screen. 80

Figure 4.26: Profile Screen. 81

Figure 4.27: Progress Screen. 82

Figure 4.28: Diet Plan Screen 83

Figure 4.29: Feedback Screen. 84

Figure 4.30: Chatbot Screen 85

Figure 5.1: System Architecture Design 86

Figure 5.2: ERD Diagram of Android-based diet consultant application 88

Figure 5.3 : Activity Diagram for Sign Up module. 91

Figure 5.4 : Activity Diagram for Sign In module. 92

Figure 5.5 : Activity Diagram for Profile module. 93

Figure 5.6 : Activity Diagram for Progress Tracker module 94

Figure 5.7 : Activity Diagram for Generate Diet Plan module. 95

Figure 5.8 : Activity Diagram for View Existing Diet Plan module 96

Figure 5.9 : Activity Diagram for Feedback module 97

Figure 5.10 : Activity Diagram for Chatbot module 98

Figure 5.11: Actual Onboarding & Navigation UIs 99

Figure 5.12 : Actual Sign Up Screen 100

Figure 5.13 : Actual Sign In screen 100

Figure 5.14 : Actual Profile Screen 101

Figure 5.15 : Actual Progress Screen 101

Figure 5.16: Actual Diet Plan Screen 102

Figure 5.17 : Actual Feedback Screen 103

Figure 5.18 : Actual Chatbot Screen 103

Figure 6.1 : Drawer and Bottom Navigation Bar 105

Figure 6.2 : Flutter Text Input Samples 105

Figure 6.3 : Loading progress bar 106

Figure 6.4 : Floating Action Button Sample 106

Figure 6.5 : Sign in with firebase authentication code segment-1 107

Figure 6.6 : Sign in with firebase authentication code segment-2 108

Figure 6.7 : Sign up with firebase authentication code segment-1 109

Figure 6.8 : Sign up with firebase authentication code segment-2 109

Figure 6.9 : Profile with firestore database code segment 111

Figure 6.10: Progress with firestore database code segment-1 111

Figure 6.11 : Progress with firestore database code segment-2 112

Figure 6.12: Progress with firestore database code segment-3 112

Figure 6.13: Generate diet plan with firestore database code segment-1 113

Figure 6.14: Generate diet plan with firestore database code segment-2 113

Figure 6.15: Store diet plan in firestore database code segment 114

Figure 6.16: Feedback in firestore database code segment 115

Figure 6.17: Example of OpenAI API key 116

Figure 6.18: Generate diet plan’s OpenAI implementation -1 117

Figure 6.19: Generate diet plan’s OpenAI implementation -2 117

Figure 6.20: Chatbot’s OpenAI implementation 118

# LIST OF SYMBOLS / ABBREVIATIONS

| | |
|---|---|
| API | Application programming interface |
| SDLC | Software development lifecycle |
| AI | Artificial Intelligence |
| FR | Functional Requirement |
| NFR | Non-Functional Requirement |
| UPF | Ultra-Processed Food |
| BMI | Body Mass Index |
| BMR | Basal Metabolic Rate |
| SVM | Support Vector Machine |
| IID | Incremental and Iterative Development |
| WBS | Work Breakdown Structure |

# LIST OF APPENDICES

Appendix A: Complete Survey Form Questions 141

Appendix B: Unit Testing 148

Appendix C: Integration Testing 158

Appendix D: User Acceptance Test 164

Appendix E: User Acceptance Test Feedback Result 176

# CHAPTER 1

# INTRODUCTION

## 1.1 Introduction

The proposed project entails the development of an Android-based Diet Consulting application in response to the prevalent health risks faced by today's generation. These health concerns necessitate the creation of a new mobile application that can offer individualized balanced diet plans tailored to each user's personal data. It's noteworthy that the World Health Organization (WHO) has classified various diseases associated with excess body mass based on extensive epidemiological studies. The mobile application under development incorporates artificial intelligence to simulate the role of a real dietitian effectively. It securely stores user information related to diet, health, and physical attributes, enabling data analysis to determine the most suitable diet plan for each user. Consequently, users of this application no longer require in-person visits to dietitians, saving them time while obtaining their required diet plans instantly. Furthermore, the application offers a comprehensive progress tracking feature by recording diet outcomes in a chart format.

The final deliverables for this project encompass an Android-based diet consultant mobile application that facilitates quick access to dietary information, enables user profile creation, provides essential food-related guidance in accordance with the diet plan, records diet outcomes for tracking progress or results, and employs artificial intelligence to evaluate the user's diet plan based on various metrics and criteria.

## 1.2 Problem Statement

Everyone always dreams to own a good health which can have a significant impact our quality of life to allow us to enjoy daily activities, achieve our life goals and lead to a fulfilling life. However, most individuals in Malaysia tends to struggle to maintain and living in a good health because there is too much distraction outside there unless that individual is very discipline and consistent and has his own plan clear in his mind. There are several boundaries of diet issues including taste preferences, personal and social factors, employment status, acculturation, and access to personal transportation and many more as this eventually causes under- or over-eating leading to consuming too much or too less nutrients per day and lacking physical exercise. In a long-term view, this will increase the risk of several diseases such as cardiovascular diseases, cancer, diabetes, and other conditions (Ramadas et al., 2021).

Malaysia has shown limited progress towards achieving the diet-related non-communicable disease (NCD) targets. 20.9% of adult women which aged 18 years and over and 15.9% of adult men are living with obesity. Malaysia's obesity prevalence is higher than the regional average of 10.3% for women and 7.5% for men. At the same time, diabetes is estimated to affect 11.9% of adult women and 13.1% of adult men (Global Nutrition Report, n.d.). Other than that, limited access of knowledge to diet or nutrition stuff also causes diet problem as well. Hence, by implementing this application in daily life, an individual is always reminded to stick to their own diet plan and able to consult for diet plan anytime or anywhere in just a tap.

### 1.2.1 Imbalanced dietary

A bad diet does not happen overnight, it is a bad habit that adopted by human gradually over time. A bad diet is about overeating or imbalanced meal whereas a most preferrable healthy diet is eating in a moderate portion of food per meal as some people tend to eat in big portions by having a lot of oily and less fibre food. Then, most people in Malaysia skips breakfast to lose weight or even to save expenses. Majority of people thought that it was a win-win situation, but this will bring opposite outcome as individuals might eat more during lunch

which led to overconsumption and hence, breakfast must not be skipped. Lastly, modern people less likely to drink sufficient water each day because of packed working schedule. Packed working schedule also makes individuals often forget about consuming sufficient water which makes the body mistake the thirst for hunger. This condition will cause people who drinks less water to feel hungry and always feels like craving for food (Othman, 2021).

### 1.2.2 Increasing indiscipline lifestyle

Maintaining a healthy and balanced diet which becoming a challenge for many individuals due to packed schedules, lack of discipline and lack of knowledge on diet. These challenges normally exist among working adults, obese person, old adults, etc. Despite the abundance of available information online, it is still difficult for people to create and follow a personalized nutrition plan that meet the individual's own preferences and health goals. As a result, many individuals struggle to make healthy food choices which leads to an increase in diet-related health problems. This problem mainly affects the following groups.

### 1.2.3 Increasing mental health issue

Peoples are always taught that having a good diet will helps in building a better physical health, but often neglect that diet will also affect mental health significantly too. Imbalance diet will lead to fatigue, impaired decision making, and can slow down reaction time. In fact, a bad diet can aggravate and may lead to major stress and depression issue as well. According to the American Dietetic Association, people tend to under- or over-eating when depressed or under stress. Poor diet when struggling with stress and depression will probably resulting in either weight gain or weight loss which is not good for the body in a long run as it may affect concentration and ability to memorize stuff (Health, n.d.).

## 1.3 Aim and Objectives

To address the issue which increasing in diet-related health problems, a diet consulting application that provide users with personalized diet plans, meal suggestions, and other features to help them achieve their health goals and make informed decisions about their diet by using Artificial Intelligence (AI) calculation algorithm.

**Objectives:**

1. To investigate diet and health related data for the proposed mobile application.
2. To conduct a business analysis based on existing application that is similar with this project.
3. To develop an android based mobile application that assist an individual to improve their diet.
4. To conduct an evaluation of the built mobile application with the target user.

## 1.4 Proposed Solution

Figure 1.1 depicted the system architecture of this application. Flutter served as the primary programming framework for this project. Development and testing were facilitated through Android Studio. The mobile application developed consisted of several features, including a progress tracker, personal profile, chatbot, generate diet plan, feedback, etc. The user's information, such as name, password, health attributes, nutrient intake, and many more, were securely stored in Firebase, a cloud-based real-time database and authentication service. Firebase offered robust data synchronization and authentication features, ensuring user data was both accessible and protected. Artificial Intelligence capabilities were leveraged through the OpenAI API. This API enabled the application to provide advanced AI-driven features, such as personalized meal planning and a virtual diet chatbot function, allowing users to interact naturally and receive health-related advice and information.

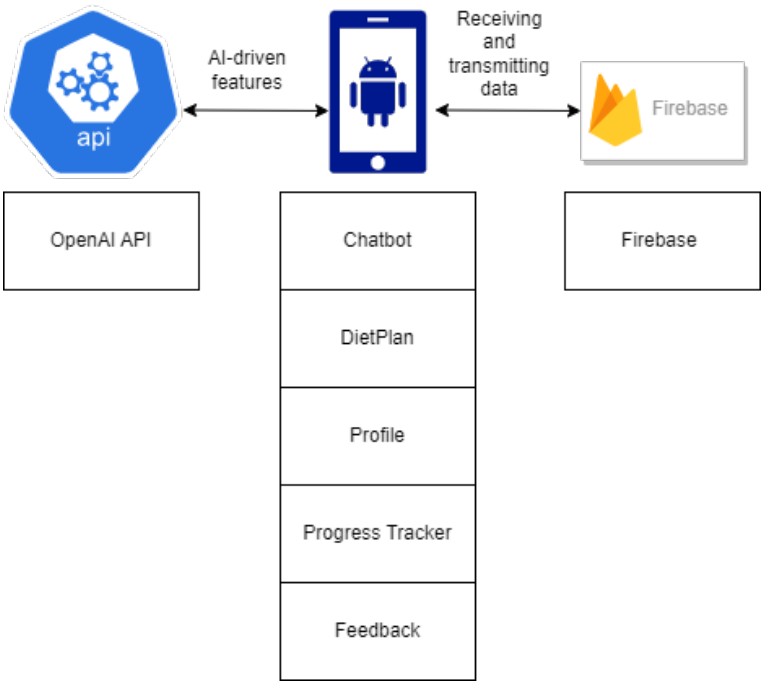



Figure 1.1.1 : System Architecture Overview

## 1.5 Proposed Approach

### 1.5.1 Research Approach

Quantitative research played an important role during data collection from a big group of diverse respondents and was implemented in the proposed project. It was commonly used to make predictions, discover facts, and test existing hypotheses among research to test theories about how or why certain events occur by finding evidence that supported or disproved the theories (Williams, 2021). The data could be clearly communicated through graphical visualization of statistics. It also provided a hard number on how the majority proportion of people thought, behaved, and felt in a certain way to back up the ideas.

The approach used in this project was questionnaires in Google Forms. The purpose of using this approach was to collect a sufficient dataset and suggestions from the target respondents. The response information of the questionnaire helped to visualize by generating a chart to be reviewed easily to gain insight into data. This enabled data exploration and communicated results to help find interesting regions and a suitable approach on certain functions. Furthermore, it was crucial to obtain user input to further support the project scope so that the application's deliverables would fulfil the user requirements.

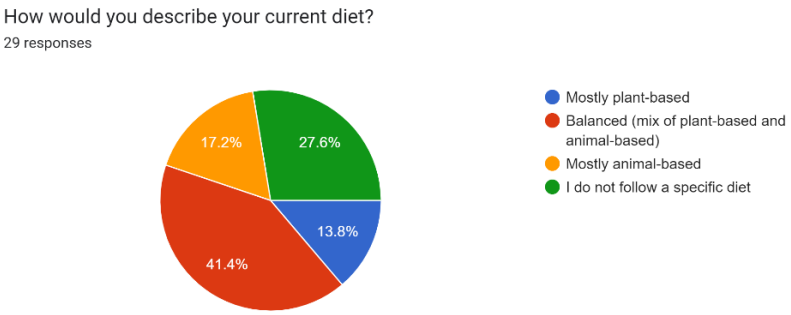



Figure 1.2 :Sample pie chart collected in Google Form.

### 1.5.2 Development Approach

For this project, an iterative and incremental development approach was selected as the preferred method of development. It was one of the types of SDLC models that constructed a partial implementation of a total system initially so that it would be in a deliverable state. Next, extra functionality was

added to increase the system's usefulness. If there were any defects that occurred before delivering the product, the defective function was then fixed, and a working product was delivered. The process was repeated throughout the entire product development process until it was completed. The repetitions of these processes were called iterations, and each iteration delivered a product increment (Anon, 2019).

The SDLC's iterative and incremental development model was preferred for this project because the initial product delivery was faster as it performed a partial implementation of a total system at the beginning and had a faster delivery of an initial product. The initial product could be tested and experienced by the target users of the system, and their feedback was used to further improve the system. If there were any changes in requirements, developers were able to repeat the development process once again, known as another iteration, to make changes according to the user requirements. The iterations continued until the final deliverable was produced, and the application was released to the market. This improved the collaboration between developers and end-users of the system to ensure the system met expectations, which eventually helped reduce the risk of project failure.

## 1.6 Project Scope

This project developed an android-based diet consulting application that adopted artificial intelligence for certain functionalities.

Firstly, the application could provide a diet plan or advice based on the user's criteria. The application was able to provide a diet plan based on the user's medical condition, food preferences, and health attributes. Medical condition referred to the illness or health problem of an individual, ranging from mild to severe. Food preferences referred to the individual's preferences on food, whether they liked or disliked certain foods, which could vary according to the individual's cultural background, genetics, experience, and more. Health attributes included characteristics of food such as vitamins, protein, fiber, etc., which contributed to an individual's health benefits. After analyzing all this data, the AI was able to prompt diet consulting advice like a real dietitian. Dietary information was shown to the user with just a click, eliminating the need to visit a dietitian physically. The dietary information was updated from time to time according to the user's criteria, as the user's medical condition and health attributes were key variables in analysing suitable dietary information.

Secondly, the application was able to generate basic food information in the diet plan. It could generate a list of healthy and delicious food recipes that were customizable based on the user's dietary needs and planning. Personalized diet meal plans were generated when the user intended to do so, and they covered the most basic meals of the day. The meal plan was fully randomized, but the food was selected based on the user's information and criteria.

Thirdly, the application was capable of recording the outcome of the diet to keep track and show the progress of the user's diet. It stored past data and visualized the trend of the user's relevant health indicators in a graph. This helped the user track their physical health and monitor their progress in achieving their health goals.

Moreover, the application protected the privacy and security of user's personal information and health data. Some users might have wanted to keep

their health data to themselves for privacy reasons. Others preferred to keep their user information, such as medical condition, nutrient intake, food preferences, and health attributes, confidential because the data was often sensitive and could be used for identity theft, fraud, or other criminal purposes.

The target users ranged from 18 to 64 years old. The reason for this choice was that the 18-24 and 25-34 age groups were often busy academically or financially and might not have emphasized maintaining or implementing a healthy lifestyle. These age groups were typically busy with packed schedules, which made it inconvenient to travel. This increased the risk of developing mental or physical health issues. Introducing this application allowed users to be reminded to monitor their current lifestyle and diet and access verified diet plans to take care of their dietary progress with just a tap.

The android version proposed for this project was Android 12. Android 12 update provided more powerful and accessible controls over how applications accessed data, including a new isolated section of the operating system that allowed artificial intelligence features to operate entirely on a device without potential network access or data exposure (Raphael, 2020).

# CHAPTER 2

# LITERATURE REVIEW

## 2.1 Introduction

Rapid and noticeable socioeconomic advancements in Malaysia over the last two decades have led to significant shifts in community lifestyles. The significant changes in the dietary patterns of Malaysia includes increased consumption of fats, oils, and refined carbohydrates. This will eventually resulting increased in percentage of fat for daily intake. Malaysia is the fattest country in Asia according to World Health Organization (WHO) in 2019. The main reason is because majority of Malaysians are having sedentary lifestyle, stress and diet which consists of large amount of ultra-processed foods (UFPs) that can be some of the reasons that led to the major problem of obesity and underweight in this country. A cross-sectional survey of dietary habits and nutrient intake discovered that a high consumption of UPFs was associated with a high BMI, a larger waist circumference (WC), and an increased risk of being obese when compared to a low consumption of UPFs (Zheng Yii, 2021). Rapid growth in industrialization and urbanization in recent decades have inevitably resulted in dietary changes of Malaysian society. Double burden of overweight and underweight situations lead to increases of getting diet-related non-communicable diseases such as hypertension, cardiovascular diseases and certain forms of cancers. In conjunction with the 12th Malaysia Plan (2021-2025) of achieving a decent standard of living, the National Plan of Action for Nutrition of Malaysia (NPANM) III (2016-2025), which is in line with the National Nutrition Policy (NNP) 2005, plays a significant role in raising whole nations' nutritional well-being through a series of appropriate strategies and activities to address this problem. Strengthening food and nutrition research and development is one of the NPANM III (2016-2025) recommended facilitating strategies for the implementation and evaluation of evidence-based nutrition programmes in Malaysia. Research and development play a vital role in food and nutrition services to keep pace with latest nutrition health care policies and practices which continuously help in improving lifestyle quality and drives

innovative technological idea related to food and nutrition (Ministry of Health Malaysia, 2020).

A diet consulting application is able to bring increase in quality of lifestyle, brings confidence on own physical attribute and to the user. A literature review of relevant topics and similar systems was undertaken to gain a better understanding of how to construct and design the application. This chapter studies on how to construct the development framework and understand on health-related calculation metrics. Furthermore, mobile applications should be evaluated to ensure that they meet the user requirements that have been developed. To build an AI chatbot that is able to provide a tailored diet consulting advice to user, several measurement methods are evaluated to identify the most suitable algorithm or calculation for this project. The main focuses of this part are:

1. To analyze the differences between dietician and nutritionist.
2. To research on the calculation's metrics and algorithms from related works.
3. To make review on similar existing diet consulting application.
4. To understand BMR, BMI, percentage of fats and muscle mass as calculation metrics from related works.
5. To understand machine learning and artificial intelligence related algorithm.
6. To review effect of diet consulting activity.
7. To analyse literature review outcomes.

## 2.2 Differences between dietitian and nutritionist

Dietitian and nutritionist both help people find the best diets and foods to meet their personal health needs, but they actually have a difference in certain areas. The simplest way to understand the distinction between dietitians and nutritionists is to consider dietetics as a specialty in addition to nutrition studies. A nutritionist is someone who uses nutrition and food knowledge to help people live healthier lives. They primarily work at the community, institutional, and population levels to provide health-promoting food information and advice. Priorities for some nutritionists have shifted with current world's changing environment into food sustainability. The focus is now on ensuring a healthy food supply that is distributed equitably across the world population in order to reduce waste and impact of global warming. Dietitians have additional qualifications to work one-on-one with patients in a clinical setting. Dietitians can work in any of the areas that nutritionist work, but dietitians can provide nutrition advice for treatment of a broad range of diseases and health conditions in addition. For example, dietitian can work clinically with individuals who have conditions such as diabetes, food allergies, gastrointestinal disorders and dietitian is able to provide helpful nutritional advice to help them improve their conditions (Anon).

## 2.3 Similar Systems Review

Diet planning is a very popular topic today among most people especially among social media influencer who wants to keep fit since smartphones and mobile applications are becoming increasingly popular in day-to-day human activities and lifestyles. This project is called an Android-based Diet Consultant which is a mobile application with artificial intelligence related to human diets. It will act as a diet consultant in real life by giving tailored diet plan to user with the support of artificial intelligence key technology. Artificial intelligence technologies are used to make decisions about diet plans based on user inputs. In this section, a comparison will be made among existing diet related applications, along with a discussion of their advantages and disadvantages. An application review matrix was built to create a checklist on the basic details of each existing application which discovered through relevant work or user reviews.

### 2.3.1 Existing Application Comparative Analysis

Currently, diet applications have become a new trend for promoting healthy lifestyles and tracking eating pattern and habits in individuals. Diet and nutrition applications have become increasingly common and are used by many individual to track their overall health, track their food intake and also to monitor their nutrient consumption. With the abundance of diet application in the market, it is critical to compare their features in order to identify their capabilities and limitations. For this comparative analysis, the selected applications were MyFitnessPal, MyNutriDiari, Lifesum, My Diet Coach and Lost it!. Five of them have some common features such as personalized meal plans, calorie and nutrient tracker, water and fitness tracker, integration with wearables devices, recipes and meal suggestions, goal tracking and AI chatbot. However certain features only available in premium version for some applications. Some figures are attached below to give a clearer illustration on how some of the features work.

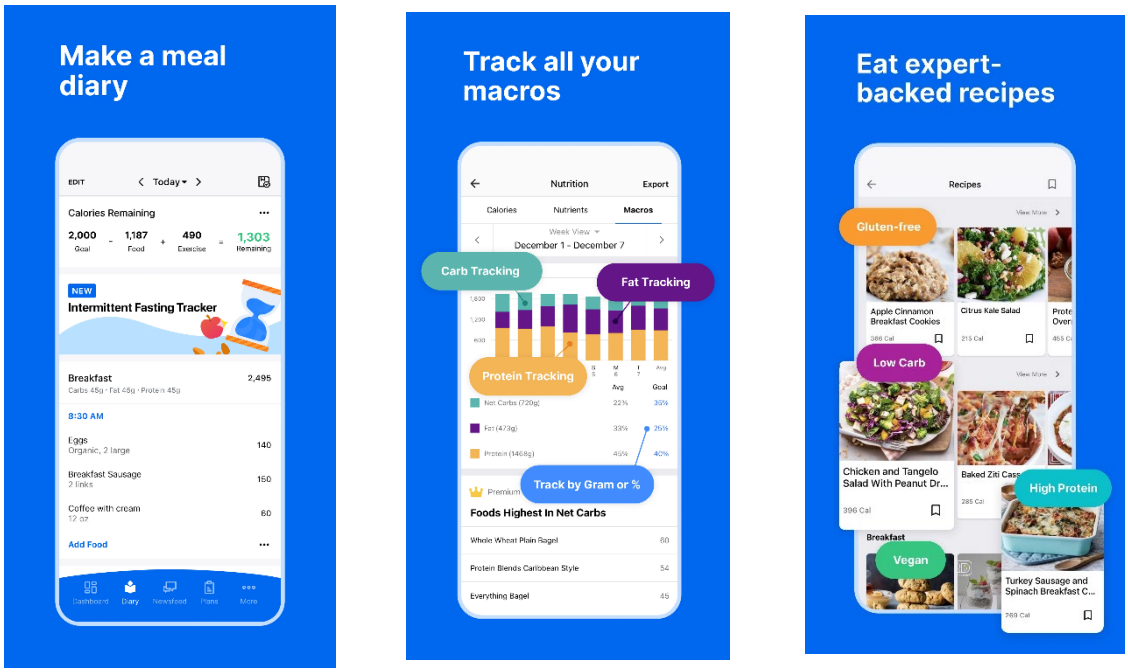



Figure 2.1: MyFitnessPal

MyFitnessPal is a popular app for tracking diet and exercise. The calorie tracking function in the application is one of the most useful features with a database of over six million food menu choices. Users can also connect

with friends for support and set weight loss or goals. MyFitnessPal can integrate with other health apps such as Fitbit, Apple Health, and Samsung Health. In short, users choose MyFitnessPal because of its matured extensive database and well-integrated with other applications and tools. Figure 2.1 shows the sample interface and screen of the application to give illustration on how the features work.

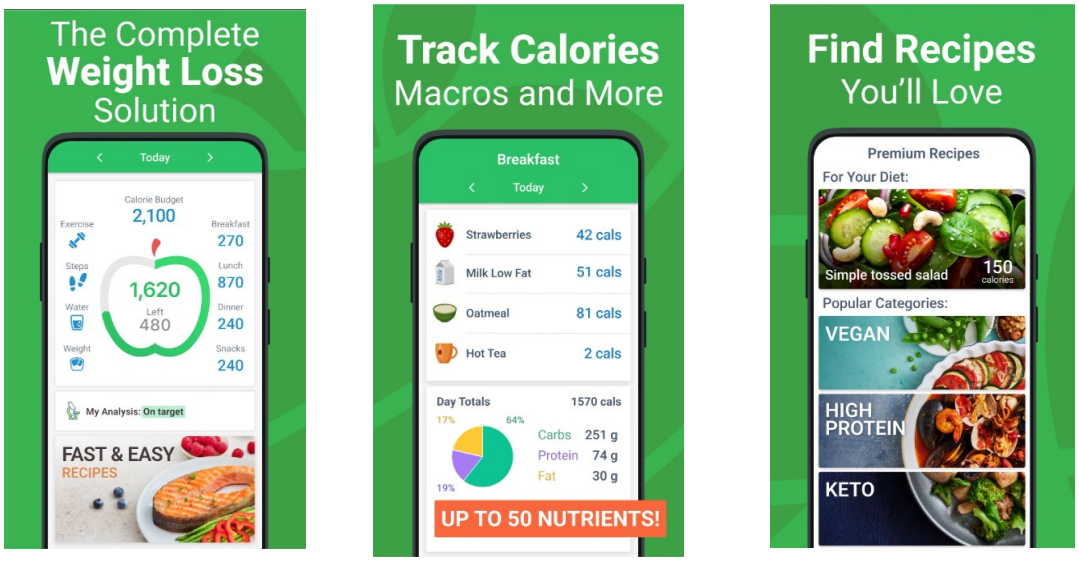



Figure 2.2: MyNetDiary

MyNetDiary is a diet-tracking app that allows users to keep track of their food intake, exercise, and weight. The application offers personalized meal plans feature and food recommendations based on the user's specific goals and dietary needs. MyNetDiary also has a barcode scanner feature that allows users to input food information easily. Users choose MyNetDiary for its personalized meal plans and barcode scanner. The interface and screen of the application are depicted in Figure 2.2, which provides a visual representation of how the features operate.

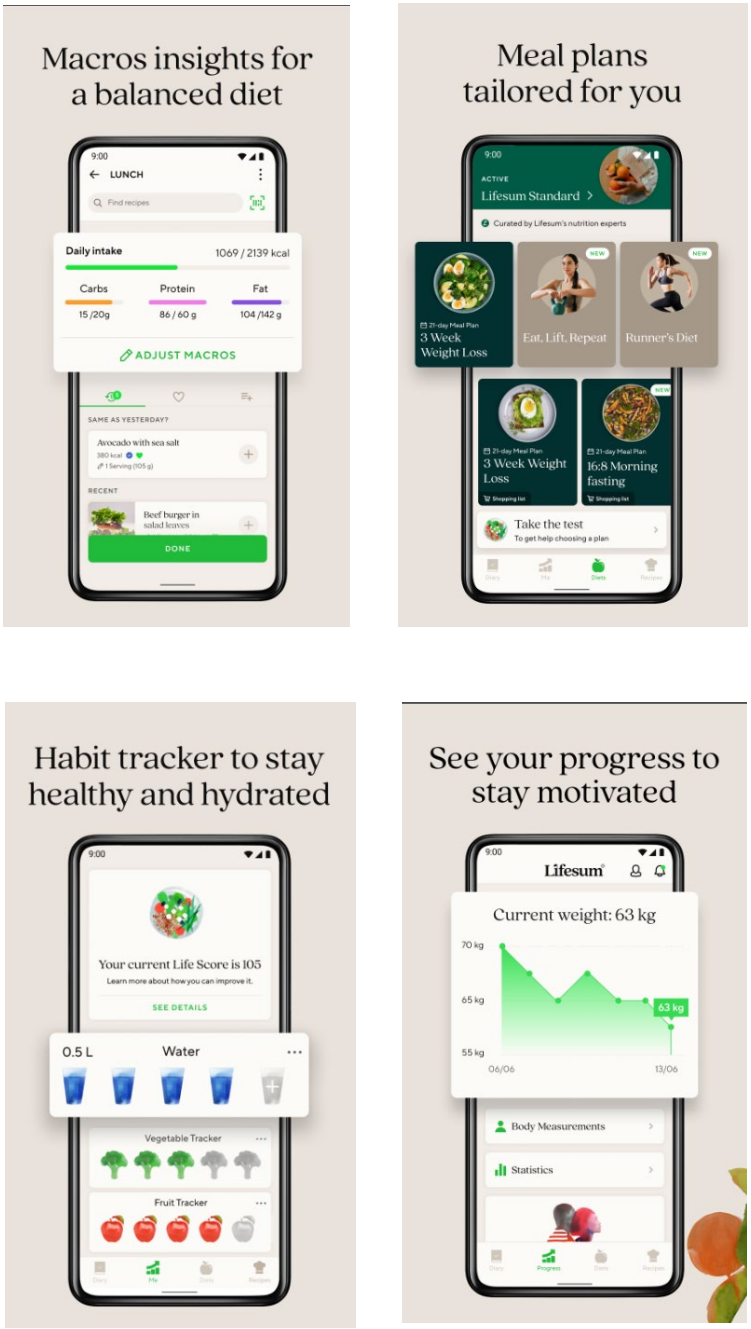



Figure 2.3: Lifesum

Lifesum is a health and fitness app that offers personalized meal plans and recipes based on the user's dietary preferences and goals. It also tracks exercise and water intake and also offers a community feature where users can connect and support each other. Lifesum's premium subscription offers even more personalized content, such as access to a personal nutritionist. Users choose Lifesum for its personalized meal plans and recipes, and the community

support feature. Figure 2.3 displays the application's interface and screen, providing a comprehensive example of how the features operate.

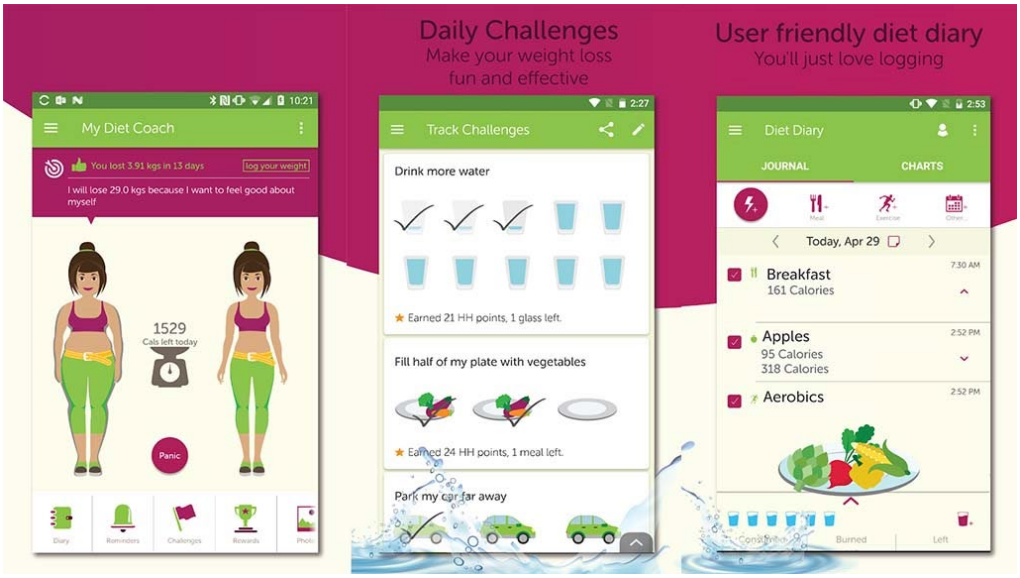



Figure 2.4: My Diet Coach (Anon, 2022).

My Diet Coach is a weight loss app that offers motivational tools such as goal tracking, reminders, and daily challenges. It also offers progress tracking and a journaling feature to help users stay accountable. My Diet Coach also provides tips and advice for healthy eating and exercise. Users choose My Diet Coach for its motivational tools and support. Figure 2.4 shows the sample interface and screen of the application to give a clearer illustration of how the screens flow.

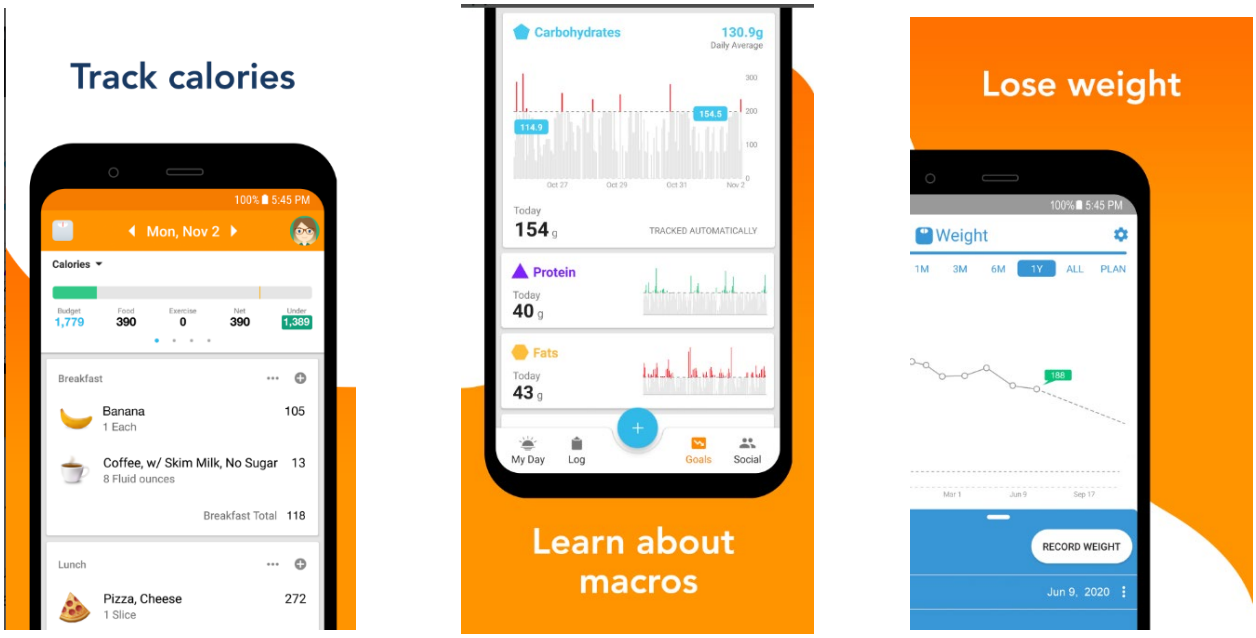



Figure 2.5: Lost It!

Lost It! is a popular application that allows users to track their food intake and exercise, set personal goals, and connect within a community. It can do pretty much everything a calorie-counting application can do. One of its unique features is the ability to create challenges with other individuals, which encourages friendly competition in burning their fats. Lost It! also can integrate with a wide range of applications and devices, such as Apple Health and Fitbit. Besides, there are more features such as a barcode scanner in the premium version. In conclusion, users choose Lost It! for its social and competitive features, and integration with other apps and devices. The sample interface and screen of the application are illustrated in Figure 2.5, which offers a more detailed view of how the features function.

The following is a summary of the feature comparison of the five existing diet-related applications that were evaluated. The common features listed in Table 1 below will be considered when determining the project scope and implementation.

Table 2.1: Features comparison among the existing similar application

| | Similar Diet Related Application | | | | | |
|---|---|---|---|---|---|---|
| Application Name / Feature | MyFitnessPal | MyNetDiary | Lifesum | My Diet Coach | Lost it! | Current Working Project |
| Personalized meal plans | ✓(Premium) | ✓(Premium) | ✓(Limited) | ✓(Limited) | ✓(Limited) | ✓ |
| Calorie tracker | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Nutrient tracker | ✓ | ✓ | ✓(Limited) | ✓(Limited) | ✓ | X |
| Water intake tracker | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Fitness tracker | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| Integration with wearable devices | ✓(Limited) | ✓(Limited) | ✓(Premium) | ✓ | ✓ | ✓ |
| Recipes and meal suggestions | ✓ | ✓ | ✓ | ✓ | ✓(Limited) | ✓ |
| Goal Tracking | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| AI Chatbot | X | X | ✓(Premium) | ✓(Premium) | X | ✓ |

After reviewing these 5 exisiting similar applications in the market, the special features of each existing similar system are evaluated and will be referred in order to assist in determining and creating the features of the proposed application. Basically, the proposed application has almost all the majority features listed in Table 1 except for nutrient tracking. This is because nutrient needs can vary greatly depending on every individual factor such as age, gender, genetics, metabolism rate, etc which make it difficult to provide personalized meal plans from the complexity of the calculation algorithms. By focusing more on the easier and simple trackable metrics such as, calorie intake

and fitness level the proposed project still can provide valuable personalized information to users. The main features, AI chatbot also available in the proposed project which is not offered in most of the existing application and usually offered only in the premium features. Hence, the proposed application will be able to bring greater value to the users.

## 2.4 Body Mass Index (BMI)

The number of calories in the food consumed in daily life determines whether a person can lose or gain weight on a diet. To lose weight, one must adjust one's diet and daily activities so that the calories used by the body outweigh the calories absorbed by the body. When the calories absorbed by the body outweigh the calories used by the body to eat, the excess calories are retained in the form of fat reserves, which impacts a person's weight gain. Hence, BMI can be related to this proposed project as it is a commonly used metric for determining an individual's weight or even health status. In the proposed project, BMI can be used as one of the calculation metrics in determining an individual's daily calorie needs and then creates a personalized diet plan from it. With the calculation of the BMI, the application can better understand to make an appropriate recommendations for each underweight or overweight user. The first step in determining the fact that a person has a healthy weight is to compute the body mass index (BMI) by using a popular formula. BMI is the ratio of weight divided by the square of one's height, which can be computed as the formula below:

$$BMI = \frac{Body\ Weight(kg)}{Body\ Height(m^2)}$$

Generally, the calculated value will be on the small side for skinny people and has greater value for obese people (Anggraeni, n.d.). Classification of value can be illustrated in the following table:

Table 2.2: BMI (Anggraeni, n.d.)

| BMI | Classification |
|---|---|
| < 14.9 | Severely Underweight |
| 15 – 18.4 | Underweight |
| 18.5 – 22.9 | Normal |
| 23 – 27.5 | Overweight |

| | |
|---|---|
| 27.6 – 40 | Obese |
| > 40 | Severely Obese |

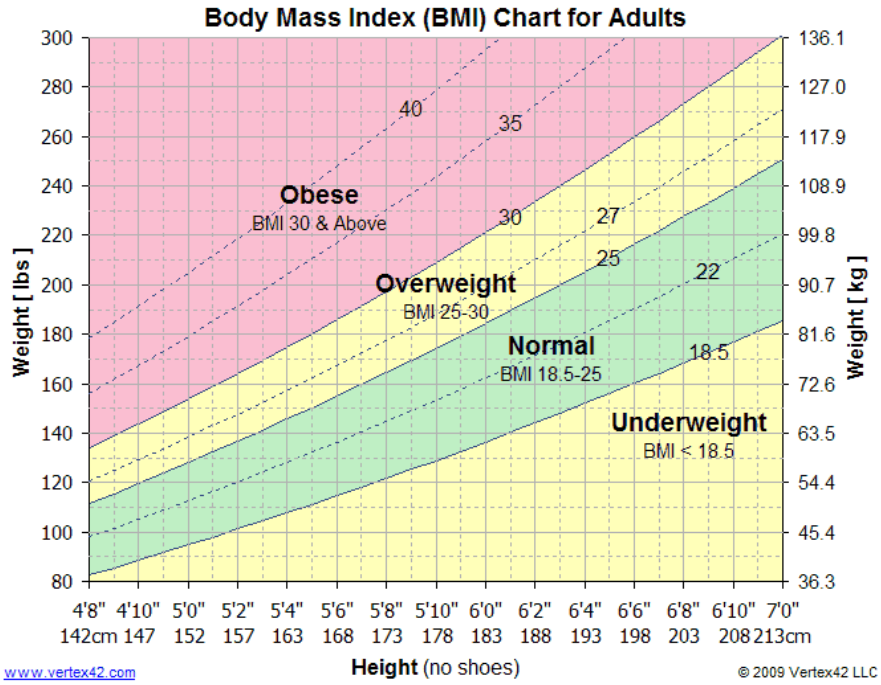



Figure 2.6: Body Mass Index (BMI) Chart for Adults (Wittwer, 2018)

A BMI Chart, also known as a Body Mass Index Chart, is a helpful tool for visualising the weight ranges for underweight, healthy weight, overweight, and obesity by referring on a person's height. Due to its simplicity, the BMI formula has grown in popularity as an initial diagnostic method for determining a person's ideal body weight. This information can help users to set goals for maintaining their diet plan or reaching their weight loss target and also provide guidance on generating the diet plan. However, the formula has flaws because it does not account for age, frame size, gender, or muscularity. For instance, BMI can be used for accessing the risk of getting obesity and encountering health issues but it is still not a perfect measure of health and may not be suitable for certain group of individuals that has high muscle mass such as athletes (Wittwer, 2018).

## 2.5 Basal Metabolic Rate (BMR)

Basal Metabolic Rate can be used to assist on gaining, losing or even maintaining a person's weight. The BMR is commonly used to calculate our body's daily calorie intake so that the application can provide a more accurate calorie recommendations for the generation of personalized diet plan. BMR is the amount of energy or calories required by the body while at rest. So, even when we do not indulge, our bodies require a certain amount of energy or calories to function properly. The number of calories or energy can only power the organ, which happens to be a vital organ in our bodies. The greater a person's muscle mass, the more energy or calories your body requires to maintain its stability. These BMR can be calculated using a variety of formulas. It utilises the Mifflin-St Jeor equation this time (Anggraeni, n.d.).

$$Male: BMR = \big(10 \times weight(kg)\big) + \big(6.25 \times height(cm)\big) - \big(5 \times age(yrs)\big) + 5$$

$$Female: BMR = \big(10 \times weight(kg)\big) + \big(6.25 \times height(cm)\big) - \big(5 \times age(yrs)\big) - 161$$

BMR calculations differ for each individual based on their level of physical activity. To obtain a much more precise calculation, the results of BMR calculation using the table below (Table 4) should be multiplied by an activity factor determined by daily activities.

Table 2.3: BMR Activity and Factor

| Activity | Activity Factor |
|---|---|
| No exercise | 1.2 |
| Light | 1.3 |
| Moderate | 1.6 |
| Heavy | 1.9 |

BMR is a crucial calculation metrics in determining an individual's daily caloric needs. By understanding an individual's BMR can help in developing a better personalized diet plan based on their daily calorie needs. A personalised diet plan can be suggested better by using an individual's BMR to help them achieve their specific goals by ensuring that they consume an appropriate number of

calories that matches their daily energy needs. In short, BMR can be applied during the creation of personalized diet plan for each individual based on their own specific metabolic rate although BMR is not the perfect indicators of health but by factoring it in this metrics, the creation of personalized diet plan will be more accurate.

## 2.6 Percentage of Fats

Body fat is an essential energy reserve in the body, which comprises adipose tissue made up of adipocytes, blood vessels, and nerves. The formation of fat cells occurs primarily during infancy and adolescence, and the number of fat cells remains relatively stable throughout adulthood. While weight gain can lead to the enlargement of existing fat cells, weight loss predominantly involves a decrease in their size rather than the total number of adipocytes. Measuring total body fat percentage is an essential health indicator since excessive body fat accumulation is associated with an increased risk of various chronic diseases, including heart disease, stroke, type 2 diabetes, and certain cancers (Forbes, n.d.).

However, these conditions can be prevented by monitoring and maintaining on to a healthy level of body fat percentage. A registered dietitian can help to design a safe, nutritious, sustainable meal plan that controlling a person’s percentage of body fats at a healthy level to meet that individual’s goal. Body fat percentage might differ for both men and women and it also reflects how physically active and fit that that person. It is a known fact that women generally tend to have a higher body fat percentage than men. This is because women's bodies have evolved to meet the unique demands of childbirth and other hormonal factors that contribute to higher body fat levels. As a result, the average body fat percentage for women is slightly higher than that of men (Forbes, n.d.). Normally, percentage of body fat can be measured in various method such as hydrostatic weighing, bioelectrical impedance analysis(BIA) scales, etc. Healthy body fat percentage ranges for both men and women which considered healthy are shown in Table 4 below:

Table 2.4: Healthy body fat percentage for men and women (Forbes, n.d.).

| Healthy Body Fat Percentage For Men and Women | | |
|---|---|---|
| Age | Body Fat Percentage for Men | Body Fat Percentage for Women |
| 20-39 | 8%-19% | 21%-32% |
| 40-59 | 11%-21% | 23%-33% |
| 60-79 | 13%-24% | 24%-35% |

The percentage of body fat is one the crucial calculation metrics on increase the accuracy of suggesting the personalized diet plan to the user of this proposed application. By knowing the percentage of the body fats of an individuals can help to understand more on an individual's fitness level and finding out the optimal body fat level. Body fat can be used to determine the appropriate calorie intake matching an individual's goals by providing a suitable a personalized diet plan. For instance, an individual that has a high percentage of body fat and aims to reduce their body weight, their calorie intake must be lower than an individual with a lower percentage of body fat. With the use of percentage of body fats as the calculation metrics, the proposed application will be able to enhance the personalized features accuracy based each individual's body fat level.

## 2.7 Muscle Mass

The weight of skeletal muscle in an individual's body, commonly referred to as muscle mass, can account for up to 40% of their total body weight. It also refers to the amount of soft muscle tissue in the body which primarily help with movement and maintaining a person's posture. A body's collective muscle tissue constitutes it's muscle mass which also refers to a common term called 'skeletal muscle'. Skeletal muscle plays a key. By taking muscle mass into the calculation metric on indicating an individual's health status. Determining an individual's muscle mass is typically more challenging than calculating their BMI. However, with advances in technology, measuring muscle mass has become increasingly accessible and convenient. For example, DEXA scan that commonly used for measuring body composition able to indicate the muscle mass in an individual (Anon, 2019). Table 5 illustrated an overall picture of an idea for muscle mass percentages for different age group on both men and women.

Table 2.5: Muscle mass percentage (Kirsten Nunez, 2019).

| Age | Muscle mass percentage averages for men (%) | Muscle mass percentage averages for women (%) |
|---|---|---|
| 18-35 | 40-44 | 31-33 |
| 36-55 | 36-40 | 29-31 |
| 56-75 | 56-75 | 27-30 |
| 76-85 | <31 | <26 |

Muscle mass can be one of the important calculation metrics for the personalized diet plan is it can indicate the amount of lean body mass an individual has. By understanding an individual's muscle mass can helps in improving the process of determining the suggestion for diet plan that is able to meets the user's energy needs while supporting muscle growth for user of the proposed application. This indicator for muscle mass can also help in tracking muscle mass from time to time to keep track on the effectiveness of the proposed diet plan by the application.

## 2.8 Rule-Based System Technique (RBS)

A rule-based system (RBS) is a type of Artificial Intelligence method that uses human-created rules to store, sort, and manipulate data. This unique attribute of imitating human intelligence involves the use of rule-based systems, which necessitate a collection of facts or a data source, along with a set of regulations to effectively manipulate that data. These rules are frequently referred to as 'If statements' because they appear to follow the pattern of 'IF X occurs THEN do Y'. Figure 2.7 depicts the operation of the rule-based system technique (Farid, 2021).

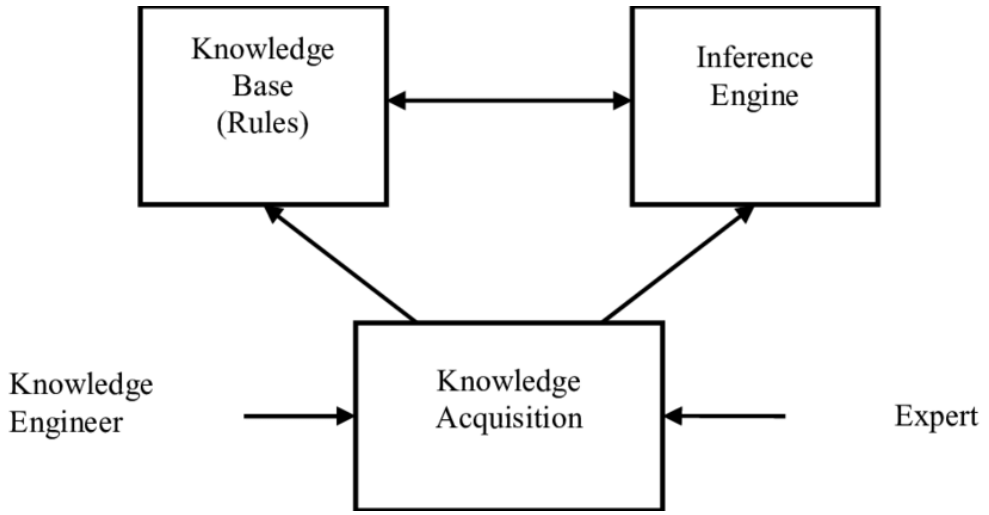



Figure 2.7 Basic architecture of rule-based system (Salah Gad Foda, 2012).

In this application, a rule-based system is expected to be used in the process recommendation when the user requests for diet plan suggestions after entering their calculated BMI, age, gender, and level of activity value. These criteria requirements are generally viewed as rules for the recommendation. These rules are used as a set of facts in which the IF statement is followed by the THEN statement, which is its outcome. The Rule-Based System technique has some advantages and disadvantages. The main advantage is that there are fewer laws to learn before understanding how the model forecasts and the way it makes sense. It's simple for basic understanding. In a given case, the mechanism can be carefully investigated, and the key variables in the forecast can be identified. By applying this technique, the application will be able to provide personalized

meal recommendations and diet plan guidance based on user's dietary needs and specific preferences. The only disadvantage is that rules-based approaches do not always produce the best estimation output. Rules-based approaches, on the other hand, are relatively simple for data with categorical attributes. The numerical aspect adds complexity because the data must be discretized or thresholds in order to define the rules (Salah Gad Foda, 2012).

## 2.9 Decision Trees

The decision tree is a well-known supervised learning algorithm utilized in machine learning to create models and make predictions regarding potential outcomes based on input data. By utilizing a tree-based representation, the decision tree algorithm is capable of resolving both classification and regression problems. In this representation, the internal nodes of the tree represent attributes, while the leaf nodes correspond to class labels. A visual depiction of this process is shown in Figure 2.8. By using decision tree, Boolean function can be represented on discrete attributes (GeeksforGeeks, 2017).

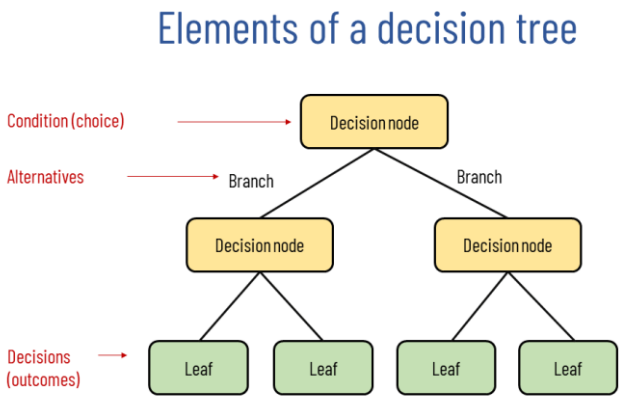



Figure 2.8 Decision Tree Element Illustration (Kosarenko, 2021)

Decision tree is often the algorithm that a data scientist started with because of decision tree's features such as interpretability, less data preparation and non-parametric. It is highly intuitive and easy to understand as the rules implemented can be visualized in a flow chart manner that can ease in solving strategic problems unlike other algorithms. However, decision trees have several

limitations which cause certain restrictions in many cases. Decision trees are high variance models and a minor change in data might cause dramatic change on the predictions produced by the model trained (Kapil, 2022). Besides, decision tree has another major limitation which is less effective in solving regression problems which is not suitable in this proposed application. In a nutshell, although the decision tree is simple, implementation of decision has its limitations which makes it not the best options to be applied in this proposed application.

## 2.10 Support Vector Machines

Support Vector Machines (SVM) is a popular pattern recognition technique utilized in various applications, including face recognition, 3D object recognition, and more. SVM encompasses two primary categories: linear separable and non-linear separable subjects. These techniques are often implemented in image processing to enhance the data's interpretability and simplify its analysis. The ultimate aim is to extract meaningful information from images for further processing and analysis (Parisa Pouladzadeh, 2012).

The data points are first represented in an n-dimensional space in this procedure. The algorithm then employs statistical techniques to choose the optimum line that separates the various classes in the data. When the data points are shown on a two-dimensional graph, the decision boundary is called a straight line. However, if there are more than two dimensions, known as hyperplanes, SVM will select the one with the greatest distance between the classes. The distance between the hyperplane and the nearest data points is defined as the margin. Both soft and hard margins are feasible. When there are two distinct sets of data and no misclassifications are acceptable, training set weights are employed during crossover to train the SVMs with a hard margin.

A soft margin is used to allow for some non-zero misclassification when greater generality from the classifier is required, or when the data cannot be neatly separated into two distinct groups (Raj, 2022).

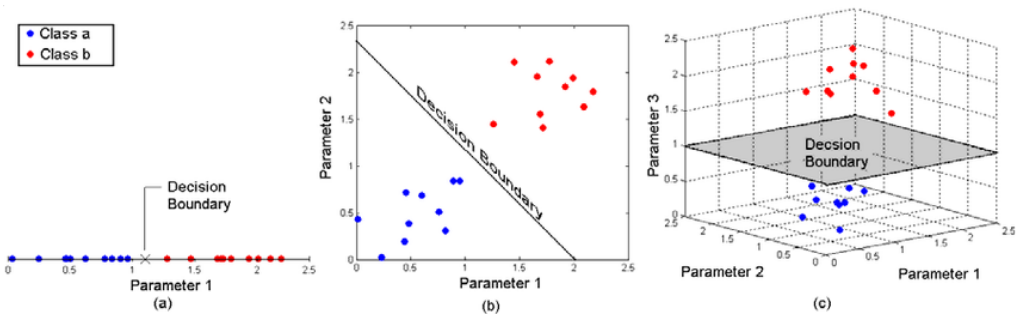



Figure 2.9: Illustration of SVM's supervised learning algorithms (Raj, 2022).

SVM is more effective in high dimensional spaces and is memory efficient. However, there are some drawbacks encountered by SVM when handling classification as the algorithms is not suitable for large data set and also especially when the data set has noise. Finally, there is no probabilistic

justification for the classification because the support vector classifier works by placing data points above and below the classifying hyperplane, making it unsuitable for use in this proposed application (Raj, 2022).

## 2.11 Summary

In a nutshell, this chapter has analyse on the comparative review by reviewing the existing similar diet related application and understand the typical or unique features that offered by each application. It is shown that each application offers it owns unique features and specialities and users have a wide option to choose from based on their own specific preferences. This analysis highlights that in order to have efficient diet goals, finding an application that suits well to one's preference is very important.

Furthermore, after making research on diet related application project, BMI and BMR are two of the most important metrics used in calculating health and fitness level of an individuals. Both of the metrics should not be used as a sole determinant for a person's health status. Additional indicator such as body fat percentage and muscle mass should be also added in the measurement as well to increase the accuracy of determining an individual's health status.

Lastly, after reviewing and looking through various diet relating work, ruled-based system technique is typically applied from a wide choices of artificial intelligence methods. This is because it allows application of a set of predefined rules to make a certain decisions and this can leads to a more accurate personalized suggestions which may able to increase user satisfaction.

# CHAPTER 3

# METHODOLOGY AND WORK PLAN

## 3.1 Introduction

This section provides a detailed explanation of the chosen development approach and its different stages, which form the basis of the project. Furthermore, the design, planning, and development techniques for the project are also discussed. To highlight the project's scope and timeline, we have constructed a Work Breakdown Structure (WBS) and a Gantt chart.

## 3.2 Incremental and Iterative Development Methodology

The incremental and iterative development (IID) approach is a software development method that emphasizes the incremental and iterative process of building software. It is actually an iterative approach, focusing on repeating the process until the desired deliverables are achieved. Each iterative will includes incremental approach, which involves adding new and additional functionality to the software in small increments (Gadam, 2023).

The IID technique breaks the software development process down into manageable parts known as iterations. Each iteration requires a specific feature set or functionality to be built, tested, and optimized before moving on to the next iteration. Iterations are typically time-boxed, meaning they have a specific start date and delivery date will be completed.

During each iteration, communication among target customer or end-user is needed to gather feedback and add changes to the product. This gathered feedback is for continuous improvement and ensures that the final deliverables meet the expectations of the customer.

The iterative nature of the IID approach allows flexibility throughout the development process. Modifications can be made to the developing software at any stage of development, allowing the software developing team to more

easily respond to changing requirements. This is in contrast to more traditional software development methodologies that follow a rigid, linear, waterfall development process (Wrike, n.d.).

The increased quality of the IID approach also allows faster deployment of functional software. Each iteration results in a functional product that the customer or end-user can test. This provides a quick response and helps ensure that the final product meets the customer's needs.

The testing and validation phase is also an important part of the IID approach. Testing is frequently made throughout the development process to ensure that the deliverable meets user's requirements, and any defects or problems are identified and addressed in the early phase.

Discussion and communication are key of the IID approach. The software development team works closely with the target customer or end user to ensure final deliverable meets the customer's needs. This helps avoid misunderstandings and ensures that the final product meets the customer's expectations. Finally, the IID approach helps lower risk by breaking down the development process into manageable chunks. This reduces the risk of project failure and allows the software development team to respond quickly to any issue.

In summary, the IID approach is a most simple and effective way to develop software that allows for continuous improvement which based on user feedback. It emphasizes collaboration, communication, testing, and integrity throughout the development process and helps manage risk by breaking down the development process into smaller manageable chunks.

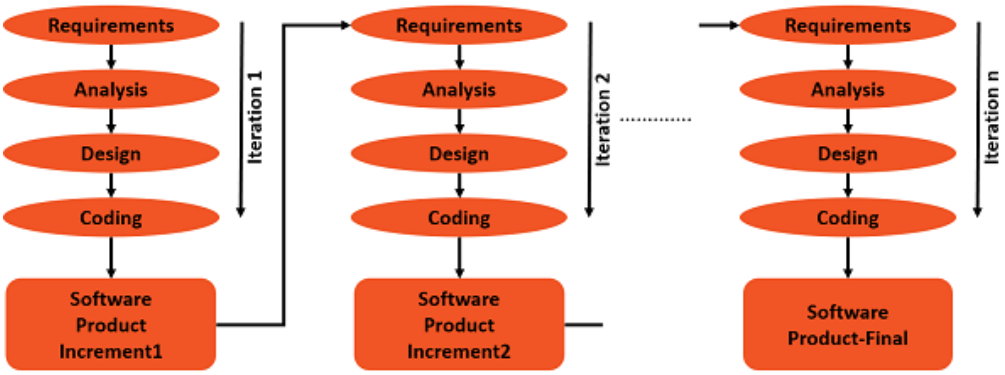



Figure 3.1: SDLC's iterative and incremental model illustration (Anon, 2019).

### 3.2.1 Planning

Every software development life cycle (SDLC) approach typically began with a planning phase. This involved outlining the project's scope and objectives, identifying the desired deliverables and requirements, and establishing a timeline to estimate the project duration. Additionally, the planning phase involved gathering and eliciting requirements to create a comprehensive list of both functional and non-functional requirements.

#### 3.2.1.1 Project Proposal

Before starting the development stage, a project proposal needed to be completed. The project title was then studied and researched in the first place about the problem commonly faced by the target user in their current dietary habit or pattern and also understanding on diet related knowledge. Several journals were viewed and properly evaluated to find out the differences between a nutritionist and a dietitian and also to compare among existing diet consultant applications to help better understand related features in similar applications. Besides, research was also made on several artificial intelligence algorithms to compare and decide the best algorithm to be used in the development phase. After determining the backdrop of the dietary issues, the problem statements were defined eventually. Following that, the project objectives were developed to specify what would be accomplished when the project was completed. The next step involved creating a project solution by examining the most suitable system architecture and selecting the development techniques that would serve

as a project guide for managing the SDLC procedures. Ultimately, it was important to define the project scope accurately, specifying the extent of the proposed application so that the final outcome met the expectations of the intended stakeholders within the given timeframe.

#### 3.2.1.2 Requirement Gathering and Elicitation

After completing and obtaining approval for the initial project planning from the project supervisor, the next phase was initiated, which involved gathering and eliciting project requirements. A questionnaire was prepared using the Google Form platform to distribute the survey form. The goal of distributing these survey forms was to gather additional data from target users about their dietary habits and expectations of a diet consultant mobile application. It took roughly 4 weeks to distribute the survey questionnaire. Once the goal number of responses was reached, the responses were examined and summarized to extract useful statistics and recommendations about the anticipated qualities of the diet consultant application.

A review of similar existing systems was also carried out in tandem with the study of the survey responses. Numerous existing systems were filtered out and chosen for research on each of their unique characteristics that could attract their users. In the literature review section, a study was conducted to understand the differences between a dietitian and a nutritionist and to gain a deeper understanding of the role of a dietitian. Furthermore, this section also included a study on various calculation metrics such as BMI, BMR, percentage of body fat, and muscle mass. Additionally, a comparison among different artificial intelligence techniques, including Rule-Based Systems, Decision Trees, and Support Vector Machines, was conducted to better understand these algorithms. After completing all requirement gathering activities, requirement elicitation continued to determine the main aspects of the proposed system and create an outline of functional and non-functional requirements.

#### 3.2.1.3 Project Scheduling

Project scheduling development was a crucial aspect of the project's planning stage. To ensure a successful outcome, it was essential to create a

comprehensive work breakdown structure (WBS) that effectively organized tasks and provided an overview of the critical functionalities required for each phase. This WBS served as a guide to ensure that all necessary tasks were identified and completed during the development process, preventing the oversight of any crucial functionalities. By utilizing a detailed WBS, project teams could effectively manage their resources and time, ultimately increasing the likelihood of project success.

One common approach to managing incremental and iterative development processes was to create a Gantt chart for each phase. This chart included projected start and finish dates for each task and helped establish the overall duration of the project. By tracking progress through the Gantt chart, project managers could ensure that each stage of development stayed on track and was completed on time. Failure to meet scheduled deadlines could result in unnecessary delays and increased costs. To maintain the project's integrity, it was crucial to adhere to both the work breakdown structure (WBS) and also the Gantt chart throughout the development process.

### 3.2.2 Analysis and Design

Once the planning phase was completed, the development process moved on to the analysis and design phase. During that stage, the system was designed based on the project scope analysis conducted in the preceding step. This involved creating multiple diagrams to provide a comprehensive understanding of the system's structure to proceed to the next step. A use case diagram was implemented, aiding in better comprehending the system's design. Furthermore, the team developed use case descriptions to provide a complete understanding of all use cases involved in the system.

In addition, to showcase the interface design and how different pages of the system interacted, interface flow diagrams and interactive prototypes were developed. This stage allowed any potential issues to be identified and addressed before moving on to the development phase. The interactive prototype solely focused on the layout and user-friendly navigation of each page,

serving as a reference for the actual application. Nonetheless, modifications could be implemented during the development process. In the subsequent chapter, detailed information regarding the interface flow diagrams and interactive prototypes of each page was presented.

### 3.2.3 Design, Development, and Testing

Following the completion of the research and design phases, the project's schedule called for the commencement of the design, development, and testing phases. The project was based on an incremental and iterative model, and some iterations were carried out before the final deliverable was created. The process of designing, developing, and testing involved breaking down user requirements into smaller sections. The crucial components were addressed in the initial iteration, while the remaining requirements were dealt with in subsequent iterations.

#### 3.2.3.1 First Iteration

In the first iteration, the emphasis was mainly on developing the user registration and login system, implementing a form for users to input their basic information, creating a database to store user profile information, and developing a user profile page to display the user's basic information.

The first step of the iteration was to research and choose a suitable authentication system to integrate into the application. Once a suitable system was selected, the system was implemented to allow users to register and log in to the application. The authentication system was then tested and refined to ensure security and ease of use.

The next step was to implement a form for users to input their basic information. The user input form was designed to collect the user's basic information, such as name, age, weight, height, medical history, etc. The user

input form was then integrated into the application and tested and refined to ensure all necessary information was collected.

The third step was to create a database to store user profile information. The database schema was designed to store user profile information, and the necessary database tables were created to store user information. The database integration was then implemented into the application and tested and refined to ensure data was properly stored and retrieved.

The final step was to develop a user profile page to display the basic information of the user. The page layout and elements were designed, and the user profile page was then implemented into the application. The user profile page was integrated with the user registration and input form, and tested and refined to ensure user information was displayed accurately.

At the end of the first iteration, the application had a functioning user registration and login system, a user input form for collecting basic user information, a database to store user profile information, and a user profile page to display the user's basic information. These features provided a solid foundation for further iterations that built upon the user profile information to generate personalized diet plans.

#### 3.2.3.2 Second Iteration

In the second iteration, the emphasis was on developing the functionality to generate personalized diet plans based on user goals and food preferences, as well as providing basic information about the foods included in the diet plan, such as calorie values.

The first step of the iteration was to research and select a suitable algorithm or system to generate personalized diet plans. Once the algorithm or system was selected, it was integrated into the application. The diet plan generation system was then tested and refined to ensure it generated accurate and appropriate diet plans.

The next step was to integrate an API that provided basic information about the foods included in the diet plan, such as calorie value. The OpenAI API was selected and integrated into the application, and the diet plan generation system was modified to retrieve the responses from the OpenAI API after prompting the necessary personal health attributes to the OpenAI API. The integration between the application and OpenAI API was then tested and refined to ensure it provided accurate and relevant information.

The final step of the iteration was to modify the diet plan page to display the user's personalized diet plan. The diet plan page layout was redesigned to accommodate the new features, which were the generate and view diet plan features. The diet plan and generate diet plan page were then tested and refined to ensure the diet plans were displayed accurately and clearly.

At the end of the second iteration, the application had the functionality to generate personalized diet plans based on user goals and preferences, as well as provide basic information about the foods included in the diet plan. These features provided a solid foundation for further iterations that built upon the diet plan information to allow progress tracking.

#### 3.2.3.3 Third Iteration

In the third iteration, the emphasis was on developing functionality that allowed users to log their lean mass and fat mass manually to help them stay on track with their health status.

The first step of the iteration was to research and select a suitable algorithm or system to track user’s health attributes. Once the algorithm or system was selected, it was integrated into the application, and the user input form was modified to collect more detailed information about the user's health attributes. The tracking system was then tested and refined to ensure it accurately tracked user’s health records.

At the end of the third iteration, the application had the functionality to allow users to log their fat mass and lean mass manually. These features enabled

users to stay on track with their diet plan and make adjustments as necessary, providing a solid foundation for further iterations that built upon this functionality.

#### 3.2.3.4 Fourth Iteration

In the fourth iteration, the focus was on integrating a chatbot feature into the diet planning and tracking application. The main objective was to enable users to interact with the chatbot and ask questions about their diet plan, receiving personalized responses based on their dietary needs, goals, and food preferences.

The iteration began with extensive research to identify the best practices and technologies for developing a chatbot system that could understand natural language, interpret user intent, and provide accurate and relevant responses. Based on this research, a design was developed for the chatbot system. Next, the chatbot system was implemented, and necessary algorithms were developed to integrate it with the existing application, including the incorporation of the OpenAI API. By integrating OpenAI API, the chatbot was capable of handling a wide range of user inputs and providing quick and accurate responses through the integration of the OpenAI API.

The integration process involved designing an intuitive interface that ensured the chatbot feature was seamlessly integrated with the application. To ensure its preciseness, the chatbot is limited to only give opinions about diet and health related information. Once the chatbot integration with OpenAI API, was complete, the feature was tested and refined to ensure that it met the user's needs and expectations. Feedback was collected from users to analyse the chatbot's performance, and necessary improvements were made. A comprehensive knowledge base was developed, incorporating information from the OpenAI API, to ensure that the chatbot provided accurate and relevant information. The benefit of OpenAI API is that the knowledge base was updated regularly so that user able to access with the latest dietary guidelines, research, and relevant information.

#### 3.2.3.5 Final Iteration

In the final iteration, the focus was on implementing a system for collecting and analysing user feedback to make improvements to the application.

A system that allowed users to provide feedback on their experience with the application was set up. This included reporting any issues they encountered, suggesting improvements, expressing their overall satisfaction, and also giving comments about the application. The feedback system was designed to efficiently collect user input.

Once the feedback system was integrated into the existing application, proper testing and refinement were carried out to ensure it functioned as intended. User feedback was carefully analysed to identify areas where improvements were needed. These improvements included bug fixes, enhancements to the user interface, and the implementation of new features based on user suggestions.

Prioritizing these improvements based on user feedback, the team worked promptly to implement them. Continuous monitoring of user feedback was essential, allowing for ongoing refinements to the application to ensure its relevance and usefulness to users.

In summary, the final iteration concentrated on enhancing the user experience by implementing a feedback system that enabled users to provide input for ongoing improvements to the application.

### 3.2.4 Closing

After the system had been developed and was ready to be delivered, user acceptance testing was applied to perform evaluation and validation for the mobile application. The testing measured the system's usability and whether or not it met the user's expectations. Testers were also needed to test out the features existing in the application. User feedback was collected after the user acceptance testing to further improvise the application. Project documentation

with a detailed description about the application was created, and after the finalization of documentation, the project was considered as done.

## 3.3 Tools Used in This Project

### 3.3.1 Axure Rp

Axure RP9 is a popular software design tool that is commonly utilized for developing prototypes, diagrams, and specifications. Its main feature is the ability to create interactive prototypes for web and mobile applications in a codeless style. It is utilizing event triggers that allow for interaction within the prototype. In this project, Axure RP9 will be utilized to create an interactive prototype for a mobile application.

### 3.3.2 Visual Studio Code and Android Studio

In the development of this web application project, Visual Studio Code served as the primary code editor. It is a versatile tool that supports numerous programming languages, including but not limited to JavaScript, C++, Node.js, and Python. Its capacity to detect cross-language reference proves helpful in ensuring coding accuracy. In addition, Visual Studio Code offers an extensive list of supported languages that can be accessed by installing relevant extensions. On the other hand, for the creation of the proposed mobile application, Android Studio utilized as the main tool to showcase the output, providing a comprehensive environment for Android app development.

### 3.3.3 Flutter

Flutter is an open-source UI framework developed by Google, ideal for creating android-based mobile applications. With Flutter, compiled applications can build natively with a single codebase by using the Dart programming language. Flutter's extensive library of widgets can be utilized for designing a visually appealing and responsive user interface and take advantage of its hot reload feature for real-time iteration. Application can be tested thoroughly, then deploy it directly to the Google Play Store. Flutter's flexibility, performance, and cross-

platform capabilities make it a powerful choice for developing this project, allowing for efficient development, maintenance, and updates.

### 3.3.4 OpenAI API

OpenAI's API is a development tool. It allows developers to integrate OpenAI's powerful natural language processing capabilities into their own applications, products, or services. With the OpenAI API, developers can build applications that generate human-like text, perform language translation, answer questions, assist with content creation, etc. It's a valuable tool for creating intelligent and language-aware applications, chatbots, virtual assistants, and other software that can understand and generate human language. This tool is used in generating user's diet plan and virtual dietitian functionality.

### 3.3.5 Enterprise Architect

Enterprise Architect is a powerful modelling and design tool used by businesses and organizations to create and manage complex systems and software applications. It supports various modelling languages such as UML, BPMN, and SysML, and provides features such as requirements management, project management, and version control. This tool used when constructing the use case diagram.

### 3.3.6 Firebase

Firebase serves as a prevalent choice for open-source database management in this project. It is employed to efficiently manage data that was initially housed in an SQLite database created by the Python script. Firebase offers a user-friendly graphical interface that simplifies the display of data in database tables. Beyond visualization, it also streamlines the tasks of adding, editing, and deleting database tables and data. The utilization of Firebase significantly enhances the development process, providing seamless monitoring and management of the application's database data.

## 3.4 Work Breakdown Structure (WBS)

0.0 Android Based Diet Consultant Application

1.0 Planning

- 1.1. Analyze the project title
- 1.2. Study background of the problem
- 1.3. Define problem statements
  - 1.3.1. Problem of imbalanced dietary
  - 1.3.2. Problem of increasing indiscipline lifestyle
  - 1.3.3. Problem of increasing mental health issue
- 1.4. Define project objectives
- 1.5. Define proposed solution
- 1.6. Define proposed approach
  - 1.6.1. Define research approach
  - 1.6.2. Define development approach
- 1.7. Define project scope
- 1.8. Requirement gathering
  - 1.8.1. Conduct Survey
    - 1.8.1.1. Generate questions for questionnaire
    - 1.8.1.2. Distribute the questionnaire
    - 1.8.1.3. Analyze and summarize the responses
  - 1.8.2. Review similar existing systems
    - 1.8.2.1. Review MyFitnessPal
    - 1.8.2.2. Review MyNetDiary
    - 1.8.2.3. Review Lifesum
    - 1.8.2.4. Review My Diet Coach
    - 1.8.2.5. Generate an application review matrix
- 1.9. Literature review
  - 1.9.1. Research of differences of dietitian and nutritionist
  - 1.9.2. Research of similar systems review
  - 1.9.3. Review of health or diet related calculation metrics
    - 1.9.3.1. Review of BMI
    - 1.9.3.2. Review of BMR
    - 1.9.3.3. Review of percentage of fats

1.9.3.4. Review of muscle mass

1.9.4. Review of artificial intelligence algorithm

1.9.4.1. Review of Rule-Based System Technique

1.9.4.2. Review of Decision Trees

1.9.4.3. Review of Support Vector Machines

1.10. Requirement elicitation

1.10.1. Choose the recommended features

1.10.2. Define the functional and non-functional requirements

1.10.3. Refine the functional and non-functional requirements

1.10.4. Finalize the functional and non-functional requirement

1.11. Project scheduling

1.11.1. Create Work Breakdown Structure

1.11.1.1. Identify the main task

1.11.1.2. Decompose the main tasks into smaller tasks

1.11.2. Create Gantt chart

1.11.2.1. Determine task dependency

1.11.2.2. Estimate task duration

1.11.2.3. Draft Gantt chart

1.11.2.4. Refine Gantt chart

1.11.2.5. Finalize Gantt chart

2.0 Analysis and design

2.1. Design use case diagram

2.2. Create use case description

2.3. Design interface flow diagram

2.4. Design prototype

3.0 Review feedback from Project I submission

4.0 Refine Project I documentation

5.0 Study the implementation framework and tools

6.0 Design, development and testing Iteration 1

6.1. Develop user registration and login system

6.1.1. Design user registration and login system

6.1.2. Develop user registration page

6.1.3. Develop user login page

6.1.4. Implement security measures

6.1.5. Test and refine user registration and login system

6.2. Implement form for user basic information

6.2.1. Design user basic information form

6.2.2. Develop user basic information input fields

6.2.3. Implement form validation

6.2.4. Test and refine user basic information form

6.3. Create database to store user profile information

6.3.1. Design user profile database schema

6.3.2. Develop user profile database tables

6.3.3. Implement database access and management

6.3.4. Test and refine user profile database

6.4. Develop user profile page

6.4.1. Design user profile page layout and components

6.4.2. Develop user profile page functionality

6.4.2.1. Edit profile information

6.4.3. Test and refine user profile page

7.0 Design, development and testing Iteration 2

7.1. Research and design algorithm for generating personalized diet plan

7.1.1. Conduct research on dietary guidelines and nutrition

7.1.2. Design algorithm for generating personalized diet plan based on user data

7.1.3. Define data points needed to generate personalized diet plan

7.2. Develop system to collect user data

7.2.1. Design user data input forms

7.2.2. Develop data input fields for collecting user data

7.2.3. Implement data validation and verification

7.2.4. Test and refine user data collection system

7.3. Implement algorithm to generate personalized diet plan

7.3.1. Implement algorithm based on design

7.3.2. Test and refine diet plan generation system

8.0 Design, development and testing Iteration 3

8.1. Research and design diet plan generation and tracking system

8.1.1. Conduct research on tracking systems

8.1.2. Design diet plan generation and tracking system

8.2. Implement system to display basic diet plan

8.2.1. Develop diet plan generation

8.2.2. Implement diet plan generation

8.2.3. Display basic diet plan information on user interface

8.3. Allow manual tracking

8.3.1. Develop manual tracking input fields

8.3.2. Implement data validation and verification

8.3.3. Store health data in user database

8.3.4. Test and refine tracking system

8.4. Test and refine the diet plan generation and tracking system

8.4.1. Conduct testing with focus groups

8.4.2. Analyze user feedback and make necessary changes to the system

8.4.3. Conduct further testing and refining to improve system

9.0 Design, development and testing Iteration 4

9.1. Research and design chatbot system

9.1.1. Research existing chatbot solutions

9.1.2. Design the chatbot feature to meet user needs

9.1.3. Determine how the chatbot will integrate with the existing application

9.2. Implement the Chatbot System

9.2.1. Develop the chatbot system using chosen technology

9.2.2. Integrate the chatbot with the existing application

9.3. Test and Refine the Chatbot Feature

9.3.1. Develop test cases to verify the chatbot's functionality

9.3.2. Test the chatbot's responses to various user questions and inputs

9.3.3. Refine the chatbot's responses based on user testing

10.0 Design, development and testing Final Iteration

10.1. Research and design user feedback function

10.1.1. Research feedback collection and analysis methods

10.1.2. Design the feedback collection and analysis system

10.2.Implement Feedback Collection and Analysis System

10.2.1. Develop the feedback collection and analysis system using chosen technology

10.2.2. Integrate the feedback system with the existing application

10.3. Test and Refine the Feedback Collection and Analysis System

10.3.1. Develop test cases to verify the system's functionality

10.3.2. Test the feedback system's collection and analysis capabilities

10.3.3. Refine the feedback system based on user testing

10.4. User Acceptance Testing

10.4.1. Engage with users to perform acceptance testing of the feedback collection and analysis system

11.0 Closing

11.1. Conduct unit tesing

11.2. Conduct integration testing

11.3. Conduct user acceptance test

11.4. Create system documentation

11.5. Finalize the project documentation

## 3.5 Gantt Chart

### 3.5.1 Overview of the Project Timeline

| Name | Duration | Start | Finish |
|---|---|---|---|
| **1.0 Planning** | 50 days? | **1/30/23 8:00 AM** | **4/7/23 5:00 PM** |
| **2.0 Analysis and design** | 11 days? | **4/3/23 8:00 AM** | **4/17/23 5:00 PM** |
| 3.0 Review feedback from Project I Submission | 3 days? | 4/18/23 8:00 AM | 4/20/23 5:00 PM |
| 4.0 Refine Project I Documentation | 7 days? | 4/21/23 8:00 AM | 5/1/23 5:00 PM |
| 5.0 Study the implementation framework and tools | 21 days? | 5/2/23 8:00 AM | 5/30/23 5:00 PM |
| **6.0 Design, development and testing Iteration 1** | 28 days? | **5/31/23 8:00 AM** | **7/7/23 5:00 PM** |
| **7.0 Design, development and testing Iteration 2** | 15 days | **7/10/23 8:00 AM** | **7/28/23 5:00 PM** |
| **8.0 Design, development and testing Iteration 3** | 11 days? | **7/31/23 8:00 AM** | **8/14/23 5:00 PM** |
| **9.0 Design, development and testing Iteration 4** | 20 days? | **8/15/23 8:00 AM** | **9/11/23 5:00 PM** |
| **10.0 Design, development and testing Final Iteration** | 10 days? | **9/12/23 8:00 AM** | **9/25/23 5:00 PM** |
| **11.0 Closing** | 25 days | **9/26/23 8:00 AM** | **10/30/23 5:00 PM** |

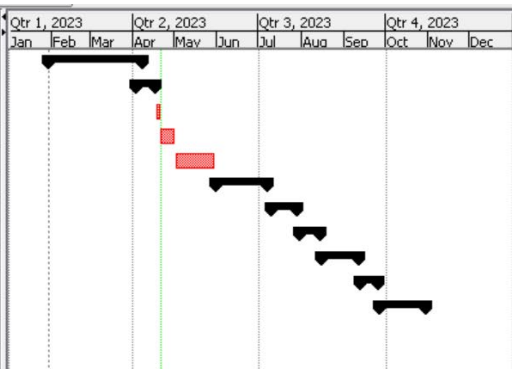



Figure 3.2: Overview of the Project Timeline.

### 3.5.2 Planning Phase Timeline

| Name | Duration | Start | Finish |
|---|---|---|---|
| **1.0 Planning** | 50 days? | **1/30/23 8:00 AM** | **4/7/23 5:00 PM** |
| 1.1 Analyze the project title | 1 day? | 1/30/23 8:00 AM | 1/30/23 5:00 PM |
| 1.2 Study background of the problem | 1 day? | 1/31/23 8:00 AM | 1/31/23 5:00 PM |
| **1.3 Define problem statements** | 3 days? | **1/30/23 8:00 AM** | **2/1/23 5:00 PM** |
| 1.3.1. Problem of imbalanced dietary | 1 day? | 1/30/23 8:00 AM | 1/30/23 5:00 PM |
| 1.3.2. Problem of increasing indiscipline lifestyle | 1 day? | 1/31/23 8:00 AM | 1/31/23 5:00 PM |
| 1.3.3. Problem of increasing mental health issue | 1 day? | 2/1/23 8:00 AM | 2/1/23 5:00 PM |
| 1.4 Define project objectives | 2 days? | 2/2/23 8:00 AM | 2/3/23 5:00 PM |
| 1.5 Define proposed solution | 2 days | 2/6/23 8:00 AM | 2/7/23 5:00 PM |
| **1.6 Define proposed approach** | 2 days? | **2/8/23 8:00 AM** | **2/9/23 5:00 PM** |
| 1.6.1. Define research approach | 1 day? | 2/8/23 8:00 AM | 2/8/23 5:00 PM |
| 1.6.2. Define development approach | 1 day? | 2/9/23 8:00 AM | 2/9/23 5:00 PM |
| 1.7 Define project scope | 3 days | 2/10/23 8:00 AM | 2/14/23 5:00 PM |
| **1.8 Requirement gathering** | 20 days? | **2/15/23 8:00 AM** | **3/14/23 5:00 PM** |
| 1.8.1. Conduct Survey | 17 days? | 2/15/23 8:00 AM | 3/9/23 5:00 PM |
| 1.8.2. Review similar exisitng systems | 3 days? | 3/10/23 8:00 AM | 3/14/23 5:00 PM |
| **1.9 Literature review** | 10 days? | **3/15/23 8:00 AM** | **3/28/23 5:00 PM** |
| 1.9.1 Research of differences of dietitian and nutritionist | 2 days? | 3/15/23 8:00 AM | 3/16/23 5:00 PM |
| 1.9.2 Research of similar systems review | 3 days | 3/17/23 8:00 AM | 3/21/23 5:00 PM |
| 1.9.3 Review of health or diet related calculation metrics | 3 days? | 3/22/23 8:00 AM | 3/24/23 5:00 PM |
| 1.9.4. Review of artificial intelligence algorithm | 2 days? | 3/27/23 8:00 AM | 3/28/23 5:00 PM |
| **1.10 Requirement elicitation** | 4 days? | **3/29/23 8:00 AM** | **4/3/23 5:00 PM** |
| 1.10.1. Choose the recommended features | 1 day? | 3/29/23 8:00 AM | 3/29/23 5:00 PM |
| 1.10.2. Define the functional and non-functional requiremer | 1 day? | 3/30/23 8:00 AM | 3/30/23 5:00 PM |
| 1.10.3. Refine the functional and non-functional requiremer | 1 day? | 3/31/23 8:00 AM | 3/31/23 5:00 PM |
| 1.10.4. Finalize the functional and non-functional requireme | 1 day? | 4/3/23 8:00 AM | 4/3/23 5:00 PM |
| **1.11 Project scheduling** | 4 days? | **4/4/23 8:00 AM** | **4/7/23 5:00 PM** |
| 1.11.1 Create Work Breakdown Structure | 2 days? | 4/4/23 8:00 AM | 4/5/23 5:00 PM |
| 1.11.2 Create Gantt chart | 2 days | 4/6/23 8:00 AM | 4/7/23 5:00 PM |

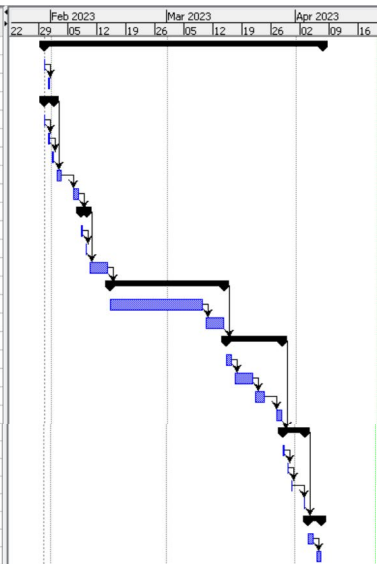



Figure 3.3: Planning Phase Timeline.

### 3.5.3 Analysis and Design Phase Timeline, and The Transition between Project I and II

| Name | Duration | Start | Finish |
|---|---|---|---|
| **2.0 Analysis and design** | 11 days? | **4/3/23 8:00 AM** | **4/17/23 5:00 PM** |
| 2.1. Design use case diagram | 2 days? | 4/3/23 8:00 AM | 4/4/23 5:00 PM |
| 2.2. Create use case description | 3 days? | 4/5/23 8:00 AM | 4/7/23 5:00 PM |
| 2.3. Design interface flow diagram | 1 day? | 4/10/23 8:00 AM | 4/10/23 5:00 PM |
| 2.4. Design prototype | 5 days? | 4/11/23 8:00 AM | 4/17/23 5:00 PM |
| 3.0 Review feedback from Project I Submission | 3 days? | 4/18/23 8:00 AM | 4/20/23 5:00 PM |
| 4.0 Refine Project I Documentation | 7 days? | 4/21/23 8:00 AM | 5/1/23 5:00 PM |
| 5.0 Study the implementation framework and tools | 21 days? | 5/2/23 8:00 AM | 5/30/23 5:00 PM |

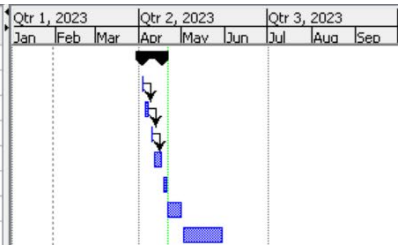



Figure 3.4: Analysis and Design Phase Timeline including the Transition between Project I and II.

### 3.5.4 Design, Development, and Testing Iteration 1 Timeline

| | Name | Duration | Start | Finish |
|---|---|---|---|---|
| 38 | **6.0 Design, development and testing Iteration 1** | 28 days? | **5/31/23 8:00 AM** | **7/7/23 5:00 PM** |
| 39 | **6.1 Develop user registration and login system** | 10 days | **5/31/23 8:00 AM** | **6/13/23 5:00 PM** |
| 40 | 6.1.1 Design user registration and login system | 2 days | 5/31/23 8:00 AM | 6/1/23 5:00 PM |
| 41 | 6.1.2 Develop user registration page | 2 days | 6/2/23 8:00 AM | 6/5/23 5:00 PM |
| 42 | 6.1.3 Develop user login page | 2 days | 6/6/23 8:00 AM | 6/7/23 5:00 PM |
| 43 | 6.1.4 Implement security measures | 2 days | 6/8/23 8:00 AM | 6/9/23 5:00 PM |
| 44 | 6.1.5 Test and refine user registration and login system | 2 days | 6/12/23 8:00 AM | 6/13/23 5:00 PM |
| 45 | **6.2 Implement form for user basic information** | 5 days? | **6/14/23 8:00 AM** | **6/20/23 5:00 PM** |
| 46 | 6.2.1 Design user basic information form | 2 days | 6/14/23 8:00 AM | 6/15/23 5:00 PM |
| 47 | 6.2.2 Develop user basic information input fields | 1 day? | 6/16/23 8:00 AM | 6/16/23 5:00 PM |
| 48 | 6.2.3 Implement form validation | 1 day? | 6/19/23 8:00 AM | 6/19/23 5:00 PM |
| 49 | 6.2.4 Test and refine user basic information form | 1 day? | 6/20/23 8:00 AM | 6/20/23 5:00 PM |
| 50 | **6.3 Create database to store user profile informatio** | 4 days? | **6/21/23 8:00 AM** | **6/26/23 5:00 PM** |
| 51 | 6.3.1 Design user profile database schema | 1 day? | 6/21/23 8:00 AM | 6/21/23 5:00 PM |
| 52 | 6.3.2 Develop user profile database tables | 1 day? | 6/22/23 8:00 AM | 6/22/23 5:00 PM |
| 53 | 6.3.3. Implement database access and management | 1 day? | 6/23/23 8:00 AM | 6/23/23 5:00 PM |
| 54 | 6.3.4 Test and refine user profile database | 1 day? | 6/26/23 8:00 AM | 6/26/23 5:00 PM |
| 55 | **6.4 Develop user profile page** | 9 days | **6/27/23 8:00 AM** | **7/7/23 5:00 PM** |
| 56 | 6.4.1 Design user profile page layout and components | 4 days | 6/27/23 8:00 AM | 6/30/23 5:00 PM |
| 57 | 6.4.2 Develop user profile page functionality | 3 days | 7/3/23 8:00 AM | 7/5/23 5:00 PM |
| 58 | 6.4.3 Test and refine user profile page | 2 days | 7/6/23 8:00 AM | 7/7/23 5:00 PM |

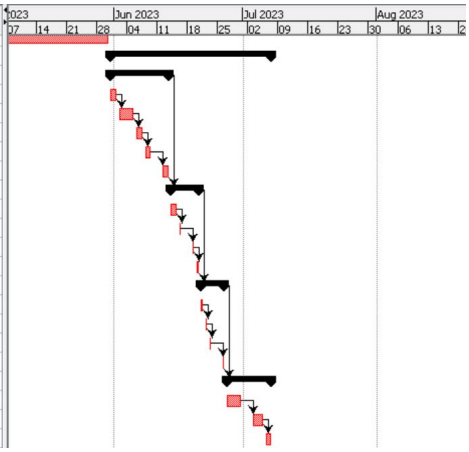



Figure 3.5: Design, Development and Testing Iteration 1 Timeline.

### 3.5.5 Design, Development, and Testing Iteration 2 Timeline

| | Name | Duration | Start | Finish |
|---|---|---|---|---|
| 59 | 7.0 Design, development and testing Iteration 2 | 15 days | **7/10/23 8:00 AM** | **7/28/23 5:00 PM** |
| 60 | **7.1 Research and design algorithm for generating pe** | 8 days | **7/10/23 8:00 AM** | **7/19/23 5:00 PM** |
| 61 | 7.1.1 Conduct research on dietary guidelines and nutrition | 2 days | 7/10/23 8:00 AM | 7/11/23 5:00 PM |
| 62 | 7.1.2 Design algorithm for generating personalized diet plan | 4 days | 7/12/23 8:00 AM | 7/17/23 5:00 PM |
| 63 | 7.1.3 Define data points needed to generate personalized d | 2 days | 7/18/23 8:00 AM | 7/19/23 5:00 PM |
| 64 | **7.2 Develop system to collect user data** | 2 days | **7/20/23 8:00 AM** | **7/21/23 5:00 PM** |
| 65 | 7.2.1 Design user data input forms | 0.5 days | 7/20/23 8:00 AM | 7/20/23 1:00 PM |
| 66 | 7.2.2 Develop data input fields for collecting user data | 0.5 days | 7/20/23 1:00 PM | 7/20/23 5:00 PM |
| 67 | 7.2.3 Implement data validation and verification | 0.5 days | 7/21/23 8:00 AM | 7/21/23 1:00 PM |
| 68 | 7.2.4 Test and refine user data collection system | 0.5 days | 7/21/23 1:00 PM | 7/21/23 5:00 PM |
| 69 | **7.3 Implement algorithm to generate personalized d** | 5 days | **7/24/23 8:00 AM** | **7/28/23 5:00 PM** |
| 70 | 7.3.1 Implement algorithm based on design | 3 days | 7/24/23 8:00 AM | 7/26/23 5:00 PM |
| 71 | 7.3.2 Test and refine diet plan generation system | 2 days | 7/27/23 8:00 AM | 7/28/23 5:00 PM |

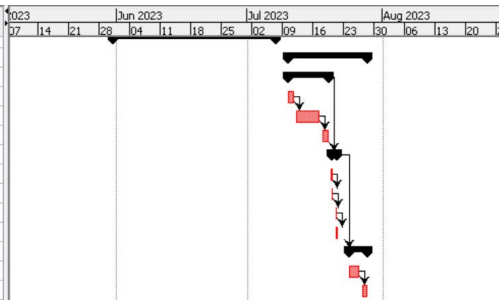



Figure 3.6: Design, Development and Testing Iteration 2 Timeline.

### 3.5.6 Design, Development, and Testing Iteration 3 Timeline

| | Name | Duration | Start | Finish |
|---|---|---|---|---|
| 72 | 8.0 Design, development and testing Iteration 3 | 11 days? | **7/31/23 8:00 AM** | **8/14/23 5:00 PM** |
| 73 | **8.1 Research and design food information and calori** | 3 days? | **7/31/23 8:00 AM** | **8/2/23 5:00 PM** |
| 74 | 8.1.1 Conduct research on nutrition and calorie tracking sys | 1 day? | 7/31/23 8:00 AM | 7/31/23 5:00 PM |
| 75 | 8.1.2 Design food information and calorie tracking system | 2 days | 8/1/23 8:00 AM | 8/2/23 5:00 PM |
| 76 | **8.2 Implement system to display basic food informa** | 2 days | **8/3/23 8:00 AM** | **8/4/23 5:00 PM** |
| 77 | 8.2.1 Develop database of food information | 1 day | 8/3/23 8:00 AM | 8/3/23 5:00 PM |
| 78 | 8.2.2 Implement search functionality for food information | 0.5 days | 8/4/23 8:00 AM | 8/4/23 1:00 PM |
| 79 | 8.2.3 Display basic food information on user interface | 0.5 days | 8/4/23 1:00 PM | 8/4/23 5:00 PM |
| 80 | **8.3 Allow manual calorie tracking** | 1 day | **8/7/23 8:00 AM** | **8/7/23 5:00 PM** |
| 81 | 8.3.1 Develop manual calorie tracking input fields | 0.25 days | 8/7/23 8:00 AM | 8/7/23 10:00 AM |
| 82 | 8.3.2 Implement data validation and verification | 0.25 days | 8/7/23 10:00 AM | 8/7/23 1:00 PM |
| 83 | 8.3.3 Store calorie tracking data in user database | 0.25 days | 8/7/23 1:00 PM | 8/7/23 3:00 PM |
| 84 | 8.3.4 Test and refine manual calorie tracking system | 0.25 days | 8/7/23 3:00 PM | 8/7/23 5:00 PM |
| 85 | **8.4 Integrate system with wearable devices to allow** | 3 days? | **8/8/23 8:00 AM** | **8/10/23 5:00 PM** |
| 86 | 8.4.1 Research and design integration with wearable device | 1 day? | 8/8/23 8:00 AM | 8/8/23 5:00 PM |
| 87 | 8.4.2 Develop integration functionality | 1 day? | 8/9/23 8:00 AM | 8/9/23 5:00 PM |
| 88 | 8.4.3 Test and refine automatic calorie tracking system | 1 day? | 8/10/23 8:00 AM | 8/10/23 5:00 PM |
| 89 | **8.5 Test and refine the food information and calorie** | 2 days | **8/11/23 8:00 AM** | **8/14/23 5:00 PM** |
| 90 | 8.5.1 Conduct usability testing with focus groups | 0.5 days | 8/11/23 8:00 AM | 8/11/23 1:00 PM |
| 91 | 8.5.2 Analyze user feedback and make necessary changes | 0.5 days | 8/11/23 1:00 PM | 8/11/23 5:00 PM |
| 92 | 8.5.3 Conduct further testing and refining to improve syste | 0.5 days | 8/14/23 8:00 AM | 8/14/23 1:00 PM |
| 93 | 8.5.4 Display basic food information on user interface | 0.5 days | 8/14/23 1:00 PM | 8/14/23 5:00 PM |

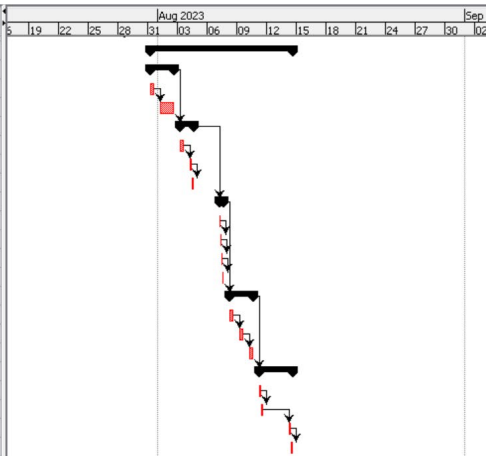
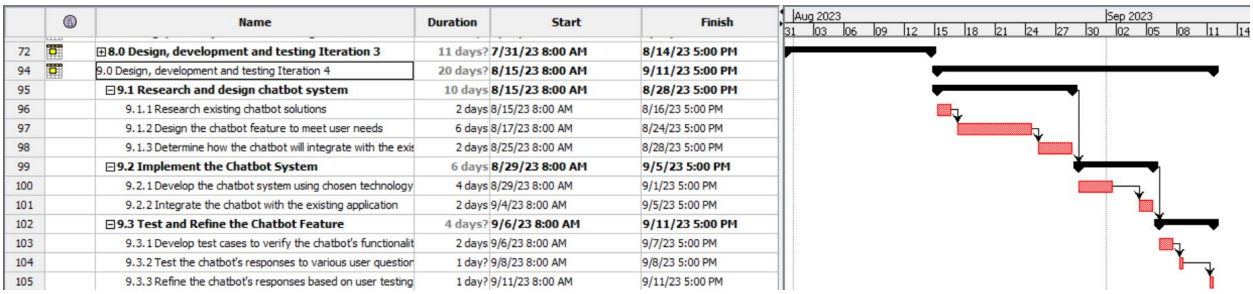



Figure 3.7: Design, Development and Testing Iteration 3 Timeline.

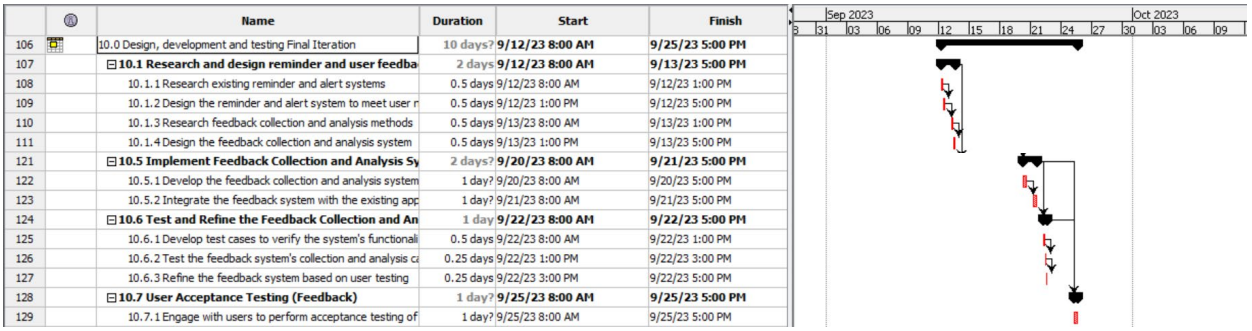
### 3.5.7 Design, Development and Testing Iteration 4 Timeline



| | Name | Duration | Start | Finish |
|---|---|---|---|---|
| 72 | **8.0 Design, development and testing Iteration 3** | 11 days? | **7/31/23 8:00 AM** | **8/14/23 5:00 PM** |
| 94 | 9.0 Design, development and testing Iteration 4 | 20 days? | **8/15/23 8:00 AM** | **9/11/23 5:00 PM** |
| 95 | **9.1 Research and design chatbot system** | 10 days | **8/15/23 8:00 AM** | **8/28/23 5:00 PM** |
| 96 | 9.1.1 Research existing chatbot solutions | 2 days | 8/15/23 8:00 AM | 8/16/23 5:00 PM |
| 97 | 9.1.2 Design the chatbot feature to meet user needs | 6 days | 8/17/23 8:00 AM | 8/24/23 5:00 PM |
| 98 | 9.1.3 Determine how the chatbot will integrate with the exis | 2 days | 8/25/23 8:00 AM | 8/28/23 5:00 PM |
| 99 | **9.2 Implement the Chatbot System** | 6 days | **8/29/23 8:00 AM** | **9/5/23 5:00 PM** |
| 100 | 9.2.1 Develop the chatbot system using chosen technology | 4 days | 8/29/23 8:00 AM | 9/1/23 5:00 PM |
| 101 | 9.2.2 Integrate the chatbot with the existing application | 2 days | 9/4/23 8:00 AM | 9/5/23 5:00 PM |
| 102 | **9.3 Test and Refine the Chatbot Feature** | 4 days? | **9/6/23 8:00 AM** | **9/11/23 5:00 PM** |
| 103 | 9.3.1 Develop test cases to verify the chatbot's functionalit | 2 days | 9/6/23 8:00 AM | 9/7/23 5:00 PM |
| 104 | 9.3.2 Test the chatbot's responses to various user question | 1 day? | 9/8/23 8:00 AM | 9/8/23 5:00 PM |
| 105 | 9.3.3 Refine the chatbot's responses based on user testing | 1 day? | 9/11/23 8:00 AM | 9/11/23 5:00 PM |

Figure 3.8: Design, Development and Testing Iteration 4 Timeline.

### 3.5.8 Design, Development, and Testing Final Iteration Timeline



| | Name | Duration | Start | Finish |
|---|---|---|---|---|
| 106 | 10.0 Design, development and testing Final Iteration | 10 days? | **9/12/23 8:00 AM** | **9/25/23 5:00 PM** |
| 107 | **10.1 Research and design reminder and user feedba** | 2 days | **9/12/23 8:00 AM** | **9/13/23 5:00 PM** |
| 108 | 10.1.1 Research existing reminder and alert systems | 0.5 days | 9/12/23 8:00 AM | 9/12/23 1:00 PM |
| 109 | 10.1.2 Design the reminder and alert system to meet user r | 0.5 days | 9/12/23 1:00 PM | 9/12/23 5:00 PM |
| 110 | 10.1.3 Research feedback collection and analysis methods | 0.5 days | 9/13/23 8:00 AM | 9/13/23 1:00 PM |
| 111 | 10.1.4 Design the feedback collection and analysis system | 0.5 days | 9/13/23 1:00 PM | 9/13/23 5:00 PM |
| 121 | **10.5 Implement Feedback Collection and Analysis Sy** | 2 days? | **9/20/23 8:00 AM** | **9/21/23 5:00 PM** |
| 122 | 10.5.1 Develop the feedback collection and analysis system | 1 day? | 9/20/23 8:00 AM | 9/20/23 5:00 PM |
| 123 | 10.5.2 Integrate the feedback system with the existing app | 1 day? | 9/21/23 8:00 AM | 9/21/23 5:00 PM |
| 124 | **10.6 Test and Refine the Feedback Collection and An** | 1 day | **9/22/23 8:00 AM** | **9/22/23 5:00 PM** |
| 125 | 10.6.1 Develop test cases to verify the system's functionali | 0.5 days | 9/22/23 8:00 AM | 9/22/23 1:00 PM |
| 126 | 10.6.2 Test the feedback system's collection and analysis ca | 0.25 days | 9/22/23 1:00 PM | 9/22/23 3:00 PM |
| 127 | 10.6.3 Refine the feedback system based on user testing | 0.25 days | 9/22/23 3:00 PM | 9/22/23 5:00 PM |
| 128 | **10.7 User Acceptance Testing (Feedback)** | 1 day? | **9/25/23 8:00 AM** | **9/25/23 5:00 PM** |
| 129 | 10.7.1 Engage with users to perform acceptance testing of | 1 day? | 9/25/23 8:00 AM | 9/25/23 5:00 PM |

Figure 3.9: Design, Development and Testing Final Iteration Timeline.

### 3.5.9 Closing Phase Timeline

| | Name | Duration | Start | Finish |
|---|---|---|---|---|
| 106 | **10.0 Design, development and testing Final Iteration** | 10 days? | **9/12/23 8:00 AM** | **9/25/23 5:00 PM** |
| 130 | 11.0 Closing | 25 days | **9/26/23 8:00 AM** | **10/30/23 5:00 PM** |
| 131 | 11.1. Conduct usability testing | 3 days | 9/26/23 8:00 AM | 9/28/23 5:00 PM |
| 132 | 11.2. Conduct user acceptance test | 5 days | 9/29/23 8:00 AM | 10/5/23 5:00 PM |
| 133 | 11.3. Create system documentation | 12 days | 10/6/23 8:00 AM | 10/23/23 5:00 PM |
| 134 | 11.4. Finalize the project documentation | 5 days | 10/24/23 8:00 AM | 10/30/23 5:00 PM |

Figure 3.10: Closing Phase Timeline.

## 3.6 Summary

The chosen approach for development was the iterative and incremental methodology. To initiate the planning stage, a planning proposal was formulated, encompassing various aspects such as the problem statement, objectives, research and development methodologies, project solution, and scope. Within the planning phase documents were the initial project requirements specification and the project schedule, which included a work breakdown structure (WBS) and a Gantt chart.

Moving into the analysis and design phase, the deliverables included the use case diagram with detailed descriptions, the interface flow diagram, and the user interface prototype design. These deliverables played a crucial role in advancing the project to the subsequent design, development, and testing phase, which consisted of five iterations to build a comprehensive project solution.

At the conclusion of each iteration within the design, development, and testing phase, a portion of the functional software system was generated. With the completion of each iteration in this phase, a fully functional software system emerged as the end product of the design, development, and testing process.

In conclusion, the project's closing phase required detailed project documentation that included all research and study findings, as well as actual deliverables from each phase of the project.

# CHAPTER 4

# PROJECT SPECIFICATION

## 4.1 Introduction

In this chapter, the initial project specification was determined using a fact-finding approach, which involved conducting a survey through Google Forms. The survey collected responses from 40 participants and identified user requirements based on the analyzed data. Next, the user requirement data was used to construct the FR and NFR. Based on functional requirements, use case diagrams and descriptions were developed, and an interface flow diagram was created. In addition, prototype screen designs were created to illustrate the overall concepts of the proposed system and demonstrate how it responded to user input.

## 4.2 Facts Finding

A survey was conducted to collect certain information from the potential users of the proposed system, which can be anyone. The questionnaire is performed by using Google Forms and the distribution method is through social media such as WhatsApp, WeChat, Instagram, etc.

40 responses were collected after several weeks of questionnaire distribution. The responses collected were then analysed and summarized to understand more about the diet pattern and knowledge of the potential users and their expectations toward a diet consultant application.

### 4.2.1 Demographics of respondents

Figure 4.1 and 4.2 summarized the age and gender of the 40 respondents. There are both 16 respondents (40%) in the age range between 18-24 and 45-64, while another 6 responses (15%) are in the age range between 25-34 and 2 respondents (5%) are in the age range between 35-44. For the gender, 24 respondents (60%) are female while 15 respondents (37.5%) are male.

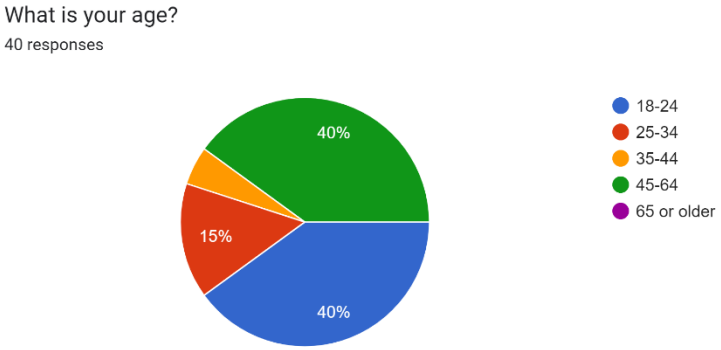



Figure 4.1: Age distribution among the 40 respondents.

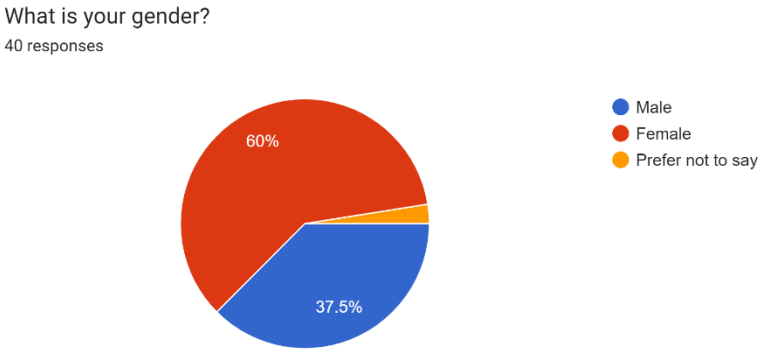



Figure 4.2: Gender of the 40 respondents

### 4.2.2 General question about user's current diet

There are 5 questions to answer by the respondents in this section. This section basically is to aim to understand the current user's diet pattern which helps to clarify the problem statements of this proposed project.

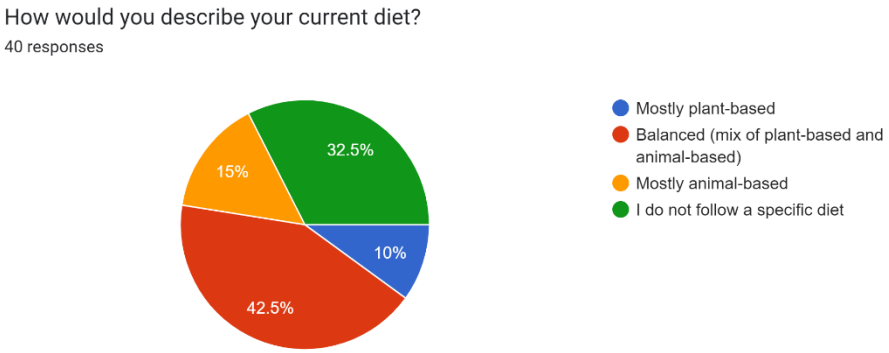



Figure 4.3: Description of 40 respondents on their current diet.

Figure 4.3 shows the description of 40 respondents on their current diet. There are 17 respondents (42.5%) that is balanced (mix of plant-based and animal-based), 13 respondents (32.5%) which do not follow a specific diet, 6 respondents (15%) state their diet are mostly animal-based while 4 respondents (10%) clarify that their diet are mostly plant-based. The reason why majority of respondents are mostly having balanced or do not have a specific diet is because they are mainly non-picky eater and does not have a specific requirement on their food preferences.

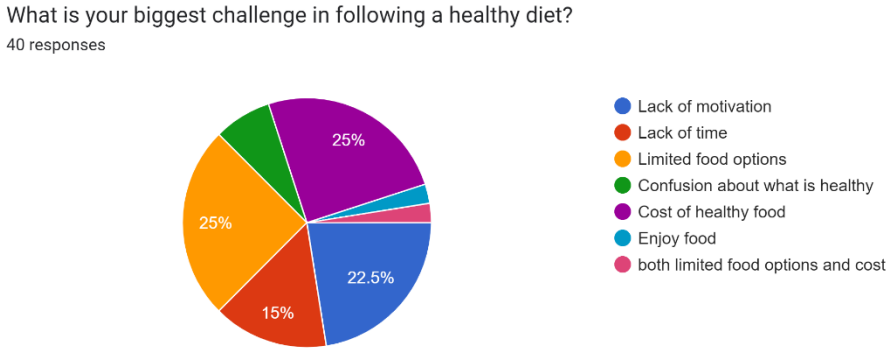



Figure 4.4: Biggest challenge of 40 respondents in following a healthy diet.

The questionnaire continues by asking for reason for biggest challenge in following a healthy diet. Figure 4.4 shows that there are both 10 respondents (25%) are limited food options and the cost of the healthy food problem. Next, there are 9 respondents (22.5%) facing lack of motivation and 6 respondents (15%) facing the problem of lack of time. This shows that, majority of the respondents want to consume healthy food, but their criteria and ability are not able to fulfil the cost of affording the healthy food which has slightly higher price than normal food.


Figure 4.5: Frequency of 40 respondents go for a diet consultation session physically.

Next, the questionnaire collects respondent's frequency of going for a diet consultation session physically. There are 32 respondents (80%) who never go for a diet consultation session physically while 7 respondents (17.5%) states that sometimes they go for a diet consultation session physically.

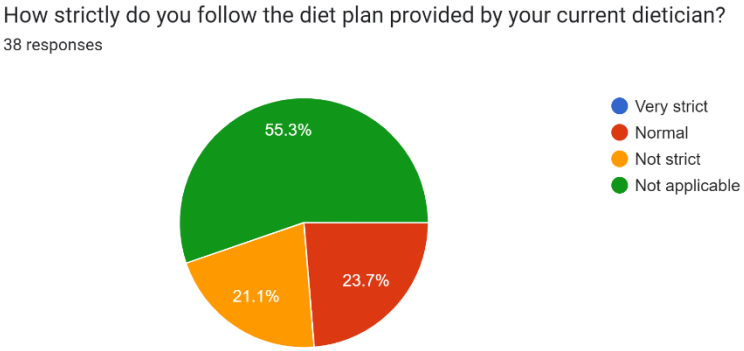



Figure 4.6: Level of strictness of the 38 respondents follow to the diet plan provided by their dietician.

Then, the questionnaire collects respondent's response on level of strictness of the 40 respondents to follow the diet plan provided by their dietician. 21 respondents (55.3%) states that this question is not applicable to them while 9 respondents (23.7%) said that they follow the diet plan provided by their current dietician in a normal manner. 8 respondents (21.2%) said that they are not strict in following the diet plan provided by their current dietician.

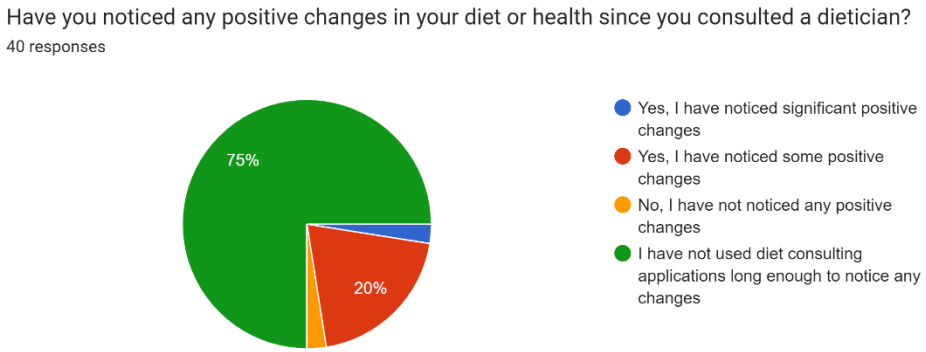



Figure 4.7: Positive changes of the 40 respondents in their diet or health after consulting a dietician.

The last part for this section of questionnaire collects the responses of the respondents on whether they noticed any positive changes in their diet or health since they consulted a dietician. There are 30 respondents (75%) which stated that they have not used diet consulting applications long enough to notice any changes while there are 8 respondents (20%) stated that they have noticed some positive changes in their diet or health after consulting a dietician.

### 4.2.3 Diet consultant application related questions among users

There are 13 questions to answer by the respondents in this section. This section basically is to aim to understand knowledges and thoughts related to diet consultant application among users to help better in specify the scope of the proposed project.

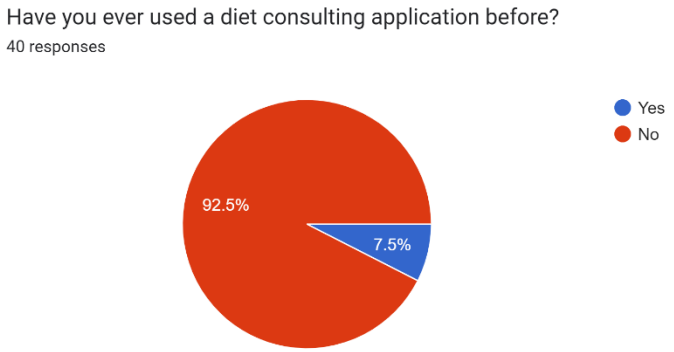



Figure 4.8: Responses of the 40 respondents on whether they ever used a diet consulting application before.

The first question for this section is about understanding the usage and experience of using diet consulting application among respondents. There are 37 respondents (92.5%) that never used a diet consulting application before while 3 respondents (7.5%) used a diet consulting application before.

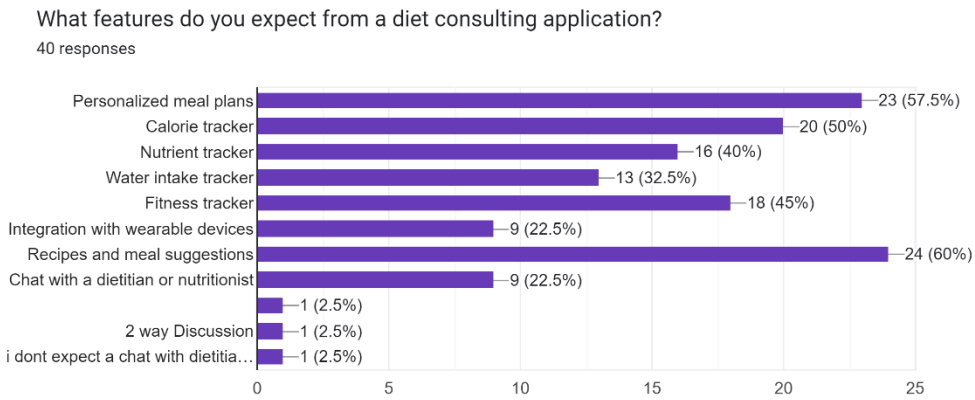



Figure 4.9: Expected features in diet consulting application from 40 respondents.

Next questions for this section is on collecting respondent's response on what features that they expect from a diet consulting application. 24 respondents (60%) voted for recipes and meal suggestions as the expected features while 23 respondents (57.5%) voted for personalized meal plans. This shows that recipes and meal suggestions and personalized meal plans will be the most expected features that should exists in a diet consulting application. Then,

there are 20 respondents (50%) voted for calorie tracker and 18 respondents (45%) voted for fitness tracker. This also shows that these respondents want to use diet consulting application as a tracker to track their own health level and fitness level to keep fit and healthy.

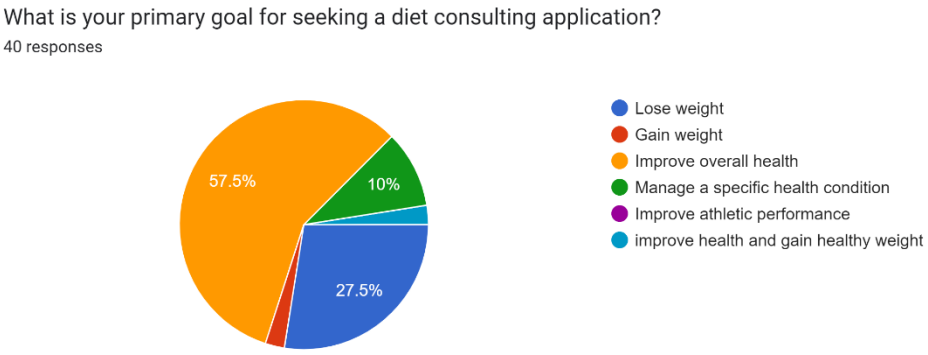



Figure 4.10: Primary goals of 40 respondents seeking from a diet consulting application.

Then, the questionnaire collected the primary goal for seeking a diet consulting application from the respondents. 23 respondents (57.5%) choose to improve overall health as their primary goal while 11 respondents (27.5%) choose lose weight as their primary goal. 4 respondents (10%) choose to manage a specific health condition as their primary goal of using a diet consulting application.

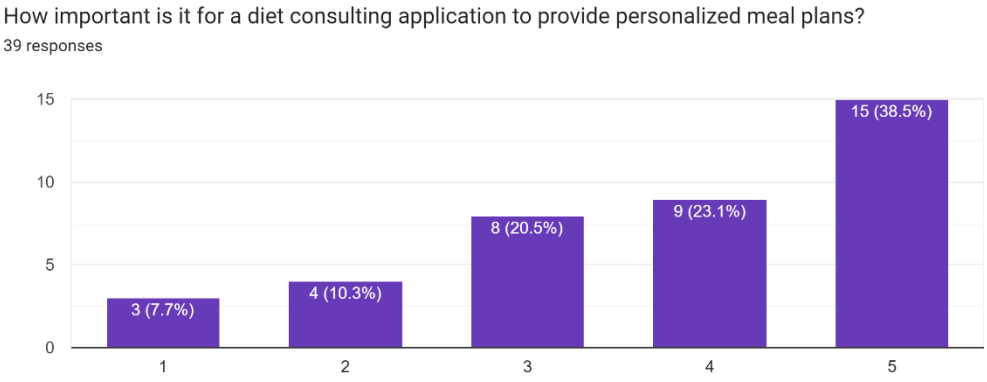



Figure 4.11: Level of importance for 40 respondents for a diet consulting application to provide a personalized meal plan.

Next, the questionnaire collects the rank of level of importance for respondents for a diet consulting application to provide a personalized meal plan. 15 respondents (38.5%) votes for importance level of 5 for a diet consulting application to provide personalized meal plans while 9 respondents (23.1%) votes for importance level of 4. In contrast, 3 respondents (7.7%) votes for importance level of 1 for a diet consulting application to provide personalized meal plans feature.

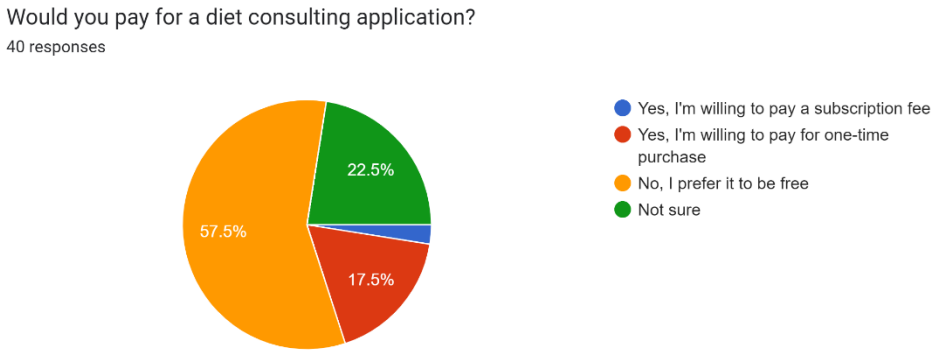



Figure 4.12: Willingness of 40 respondents to pay for a diet consulting application.

Furthermore, the questionnaire collects the willingness of 40 respondents to pay for a diet consulting application. According to the responses, 23 respondents (57.5%) say that they prefer the diet consulting application to be freee while, 9 respondents (22.5%) say that they are not sure whether they are willing to pay for the application. Then, 7 respondents (17.5%) say that they are willing to pay for the application for one-time purchase.

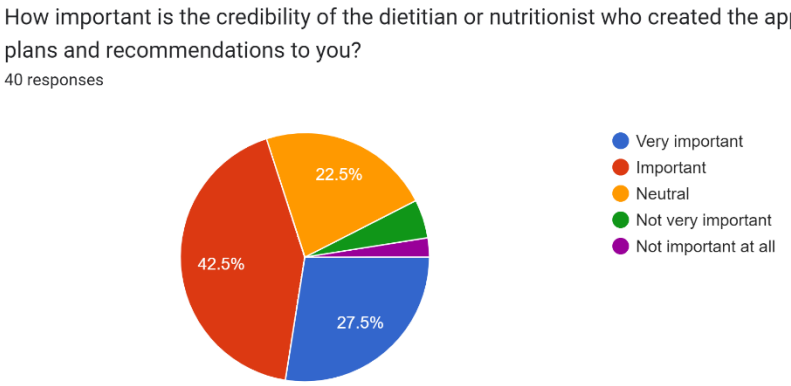



Figure 4.13: Importance of credibility of the dietitian or nutritionist who created the application's meal plans and recommendations.

The questionnaire further asked the respondents about the importance of credibility of the dietician or nutritionist who created the application's meal plans and recommendations. 17 respondents (42.5%) responded with it is important to have a high credibility dietitian to provide those recommendations. Then, 11 respondents (27.5%) state that it is very important whereas 9 respondents (22.5%) state that they are neutral to this statement.

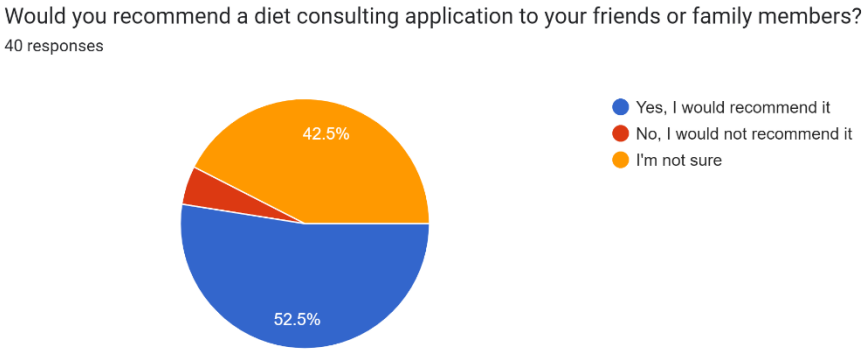



Figure 4.14: Respondent's likeliness of recommendation of diet consulting application to their friends or family members.

The questionnaire continued to ask on respondent's likeliness of recommendation of diet consulting application to their friends or family members. 21 respondents (52.5%) state that they would recommend a diet consulting application to their friends or family members if they found out that the application is useful in the first place. Then, 17 respondents (42.5%) stated

that they are not sure whether they will recommend a diet consulting to their friends or family.

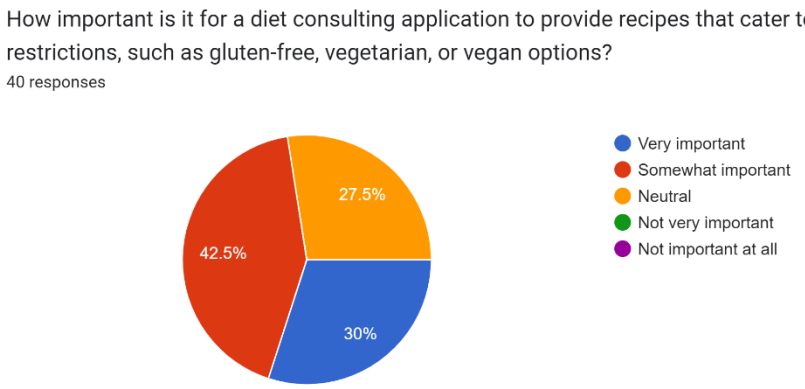



Figure 4.15: Importance of a diet consulting application to provide recipes.

The next question for this section is about the importance of diet consulting application to provide recipes that cater to dietary restrictions. 17 respondents (42.5%) stated that it is somewhat important to have this function, 12 respondents (30%) stated that it is very important to gave this function while 11 respondents (27.5%) stated that they are having neutral thoughts on this statement.

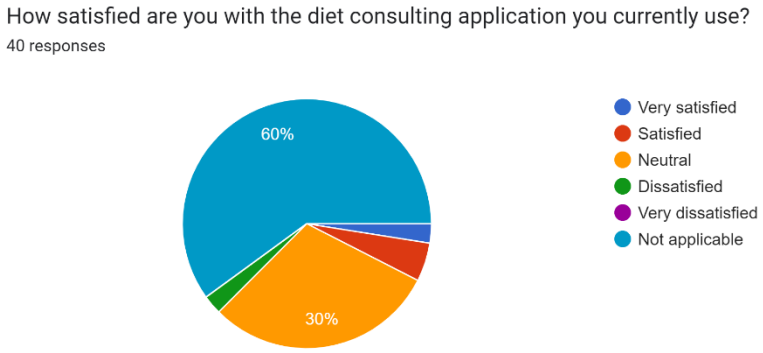



Figure 4.16: Satisfaction with currently use diet consulting application.

Next, the questionnaire asked the respondents whether they are satisfied with currently use diet consulting application. 24 respondents (60%) said that this is not applicable to them, while 12 respondents (30%) stated that

they are having a neutral opinion on this. This shows that the majority of the respondents do not have the habit of using a diet consulting application.

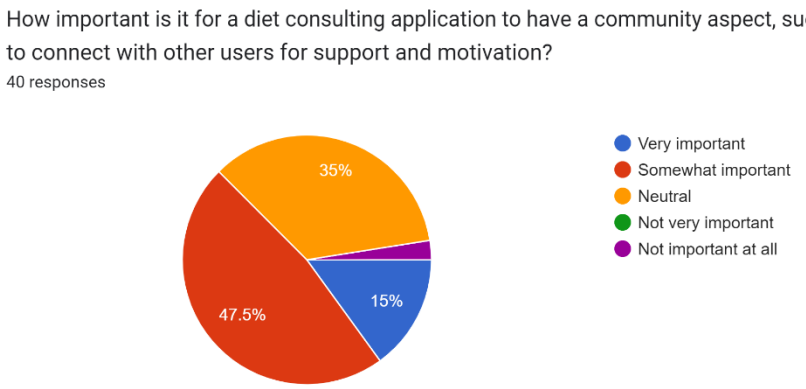



Figure 4.17: Importance of diet consulting application to have a community aspect.

The next question asked about the importance of diet consulting applications to have a community aspect. 19 respondents (47.5%) stated that the community aspect is somewhat important, 14 respondents (35%) stated that they are having neutral opinion on this aspect whereas 6 respondents (15%) stated that this community aspect is very important for a diet consulting application as the ability to connect with other users can help to increase motivation.

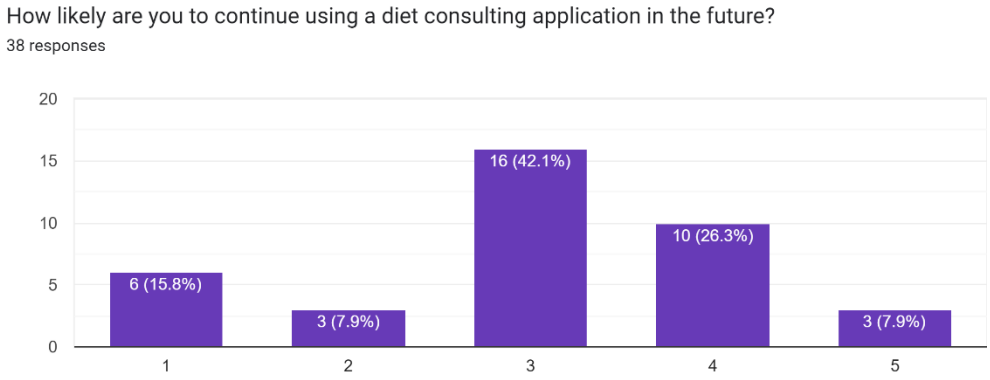



Figure 4.18: Likeliness of continuous usage of diet consulting applications in the future.

The questionnaire further asked about the likeliness of respondents' continuous usage of diet consulting applications in the future. 16 respondents (42.1%) voted for likeliness of 3, 10 respondents (26.3%) voted for likeliness of 4. In contrast, Likeliness 2 and 5 were both voted by 3 respondents (7.9%). This shows that these respondents are not sure whether they should or not to continue using diet consulting applications in the future, a promising functional diet consulting application must be introduced to keep these respondents to remain as user.

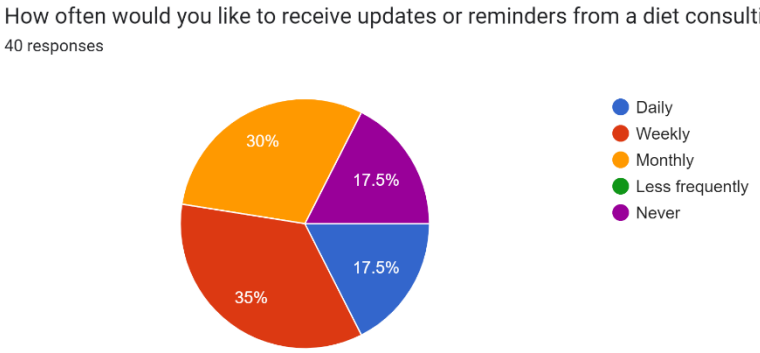



Figure 4.19: Likeliness of 40 respondents receiving updates or reminders from a diet consulting application.

Next, the questionnaire continues to ask the respondents about the likeliness of receiving updates or reminders from a diet consulting application. 14 respondents (35%) prefer to receive updates or reminders in weekly basis. 12 respondents (30%) prefer to receive updates or reminders in monthly basis. However, 7 respondents (17.5%) prefer to receive updates or reminder in both daily basis and never receive any updates or reminders.

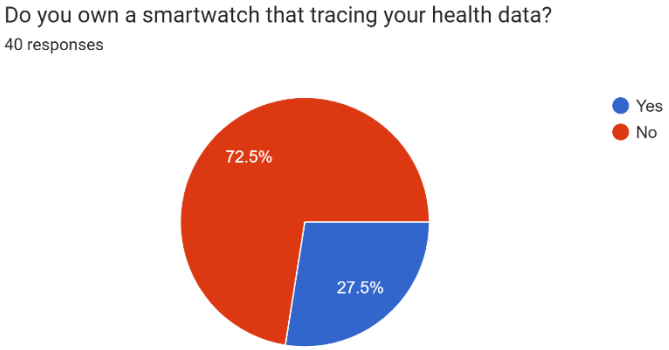



Figure 4.20: Availability of smartwatch among 40 respondents.

Lastly, the respondents were asked that whether do they own a smartwatch that able to trace their health data. 29 respondents (72.5%) stated that they do not own a smartwatch that able to trace health data while 11 respondents (27.5%) stated that they own a smartwatch that able to trace their health data.

## 4.3 Requirement Specification

There are two types of requirements specifications which is functional requirements and the non-functional requirements. The functional requirements feature that must be present in the proposed system in order fulfil the expectation of the user. Non-functional requirements of this proposed system, on the other hand, include usability, performance, availability, security, compatibility, and operational. The proposed system should meet the non-functional requirements listed so that the finalised software system has the quality and high performance.

### 4.3.1 Functional Requirement

| Functional Requirement ID | Functional Requirement Statement |
|---|---|
| FR001 | The application shall be able to allow user to register and create a user profile by providing basic information. |
| FR002 | The application shall allow user to modify their user profiles. |
| FR003 | The application shall be able to generate a personalized diet plan based on user's goals and food preferences. |
| FR004 | The application shall provide basic information about the foods included in the diet plan such as calorie value. |
| FR005 | The application shall allow users to record their progress toward their goals by tracking their health attributes. |
| FR006 | The application shall provide visual representations of progress, such as graphs. |
| FR007 | The application shall be able to allow users to log their lean mass, fat mass, height and weight manually in daily basis. |
| FR008 | The application shall have a chatbot feature that allows users to ask questions about diet related problems and receive personalized responses. |
| FR009 | The application shall be able to receive feedback from user to further improvise on the application. |

### 4.3.2 Non-Functional Requirement

1.0 Usability

1.1. The application shall be user-friendly, easy to navigate and has a clean interface so that it is easy for user to interact with the application.

1.2. The application shall be able to have easy-to-understand guideline so that user able to perform actions without guidance of a technical person.

2.0 Performance

2.1. The application shall be able to respond to the user input in less than 10 seconds.

2.2. The application shall be able to has quick load times while handle large amounts of data.

3.0 Availability

3.1. The application shall be available all the time with minimal downtime for maintenance or updates that will causes the application to become unavailable to the user.

4.0 Security

4.1. The application shall not allow user to access the account without a valid user credential to prevent unauthorized access.

4.2. The application shall protect user data privacy and security such as personal information, health data and password protection.

5.0 Compatibility

5.1. The application shall be able to run smoothly on Android version 12.0 and above.

5.2. The application shall be able to work smoothly on Android devices of various resolutions and screen sizes.

6.0 Operational

6.1 The application shall be able to has an AI algorithm that provide accurate results with a minimum accuracy rate of 90%.

## 4.4 System Use Case

### 4.4.1 Use Case Diagram

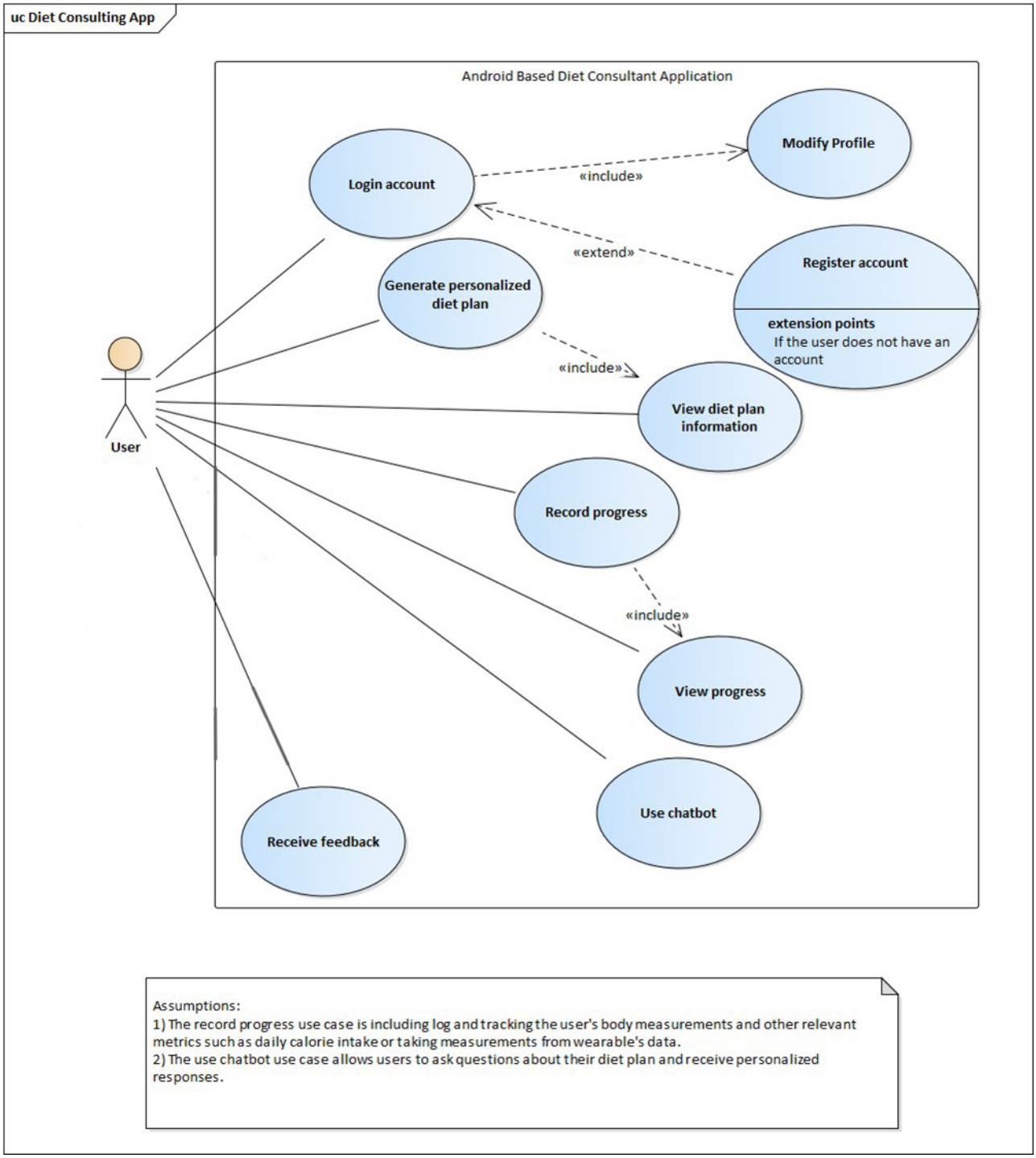



Figure 4.21: Use case diagram for the proposed application

### 4.4.2 Use Case Description

<table>
<tr><td colspan="2">Use Case Name: Login Account</td><td>ID:<br>UC001</td><td>Importance Level: High</td></tr>
<tr><td colspan="2">Primary Actor: User</td><td colspan="2">Use Case Type: Detail, Real</td></tr>
<tr><td colspan="4">Stakeholders and Interests:<br><br>User – wants to log in into his/her account to access the android mobile application.</td></tr>
<tr><td colspan="4">Brief Description: This use case describes how the user can log in to his/her account in order to access the android mobile application.</td></tr>
<tr><td colspan="4">Trigger: User wants to log in to his/her account to access the android mobile application.</td></tr>
<tr><td colspan="4">Relationships:<br>Association : User<br>Include : Modify Profile<br>Extend : Register Account<br>Generalization: -</td></tr>
<tr><td colspan="4">Normal Flow of Events:<br>1. The user launches the diet consultant application.<br>2. The user selects the login function in the welcome screen.<br>3. The user enters his/her credential in the Sign In screen.<br>4. The application validates the email address and password.<br>5. The user is successfully logged into the diet consultant application to perform the functions provided by the application.</td></tr>
<tr><td colspan="4">Sub-flows: -</td></tr>
<tr><td colspan="4">Alternate/Exceptional Flows:<br>2a: If the user does not have an account, “Register account” use case (ID:2) is performed.<br>4a: If the user enters the invalid email address or password, the application will prompt an error message and the user will need to re-enter the email address and password.</td></tr>
</table>

| Use Case Name**: Register Account** | ID**:** **UC002** | Importance Level**: High** |
|---|---|---|
| Primary Actor**: User** | Use Case Type**: Detail, Real** | |
| Stakeholders and Interests**:**<br><br>**User – wants to register an account to be used to log in and access the android mobile application.** | | |
| Brief Description**: This use case describes how the user can register an account to be used to login and access the android mobile application.** | | |
| Trigger**: User wants to register an account to login and access to the android mobile application.** | | |
| Relationships:<br>Association **: Login Account (ID: 1)**<br>Include **: -**<br>Extend **: -**<br>Generalization**: -** | | |
| Normal Flow of Events**:**<br>**1. The user launches the diet consultant application.**<br>**2. The user selects the register function in the welcome screen and redirect to Register Screen.**<br>**3. The user enters his/her age.**<br>**4. The user enters his/her weight and height.**<br>**5. The user enters his/her medical history and food preferences.**<br>**6. The user enters his/her nickname, email address and password in the Register Account**<br>**7. The user will redirect to the Home screen of the application.** | | |
| Sub-flows**: -** | | |
| Alternate/Exceptional Flows:- | | |

| Use Case Name: **Modify Profile** | ID: **UC003** | Importance Level: **Moderate** |
|---|---|---|
| Primary Actor: **User** | Use Case Type: **Detail, Real** | |
| Stakeholders and Interests:<br><br>**User – wants to update his/her profile personal information to the current latest health conditions such as medical conditions and food preferences.** | | |
| Brief Description: **The use case describes how the user can update his/her profile personal information to the current latest health conditions such as medical conditions and food preferences.** | | |
| Trigger: **User wants to update his/her profile personal information to the current latest health conditions such as medical conditions and food preferences.** | | |
| Relationships:<br>Association : **Login Account (ID: 1)**<br>Include : **-**<br>Extend : **-**<br>Generalization: **-** | | |
| Normal Flow of Events:<br>1. **The user selects on the Edit icon button besides the particular information that he/she wishes to make changes in the Profile screen.**<br>2. **The user enters the new information.**<br>3. **The user clicks the Save button the save the changes made.**<br>4. **The user will redirect back to the Profile screen.** | | |
| Sub-flows: **-** | | |
| Alternate/Exceptional Flows:<br>**3a: If the user does not make any changes, he/she can return back to the Profile screen by select on the Back button instead of the Save button.** | | |

| Use Case Name**: Generate personalized diet plan** | ID**: UC006** | Importance Level**: High** |
|---|---|---|
| Primary Actor**: User** | Use Case Type**: Detail, Real** | |
| Stakeholders and Interests**:**<br><br>**User – wants to generate a personalized diet plan based on his/her's goals, preferences, and dietary needs.** | | |
| Brief Description**: The use case describes how the user generate a personalized diet plan based on his/her's goals, preferences, and dietary needs.** | | |
| Trigger**: User wants to generate a personalized diet plan based on his/her's goals, preferences, and dietary needs.** | | |
| Relationships:<br>Association **: User**<br>Include **: View diet plan information**<br>Extend **: -**<br>Generalization**: -** | | |
| Normal Flow of Events**:**<br>1. **The user wishes to generate a personalized diet plan.**<br>2. **The application calculates the user's BMR, BMI, percentage of fats and muscle mass based on user's information.**<br>3. **The user set a goal such as losing weight, maintaining weight, or gaining weight.**<br>4. **The application generates a personalized meal plan after selecting on the 'Generate' button.**<br>5. **The user will redirect to the Diet plan view screen.**<br>6. **'View diet plan information' use case (ID:5) flow will be performed.** | | |
| Sub-flows**: -** | | |
| Alternate/Exceptional Flows:<br>**4a: If the user does not want to generate any diet plan, he/she can return back to home page by selects on 'Back' button instead of 'Generate' button.** | | |

| Use Case Name**: View diet plan information** | ID**: UC007** | Importance Level**: High** |
|---|---|---|
| Primary Actor**: User** | Use Case Type**: Detail, Real** | |
| Stakeholders and Interests**:**<br><br>**User- wants to view his/her's diet plan information after generating his/her personalized diet plan.** | | |
| Brief Description**: The use case describes how the user can view his/her's diet plan information after generating his/her personalized diet plan.** | | |
| Trigger**: User wants to view his/her's diet plan information after generating his/her personalized diet plan.** | | |
| Relationships:<br>Association **: User**<br>Include **: -**<br>Extend **: -**<br>Generalization**: -** | | |
| Normal Flow of Events**:**<br>**1. User wishes to view his/her current diet plan information.**<br>**2. User navigates to the "View Diet Plan" screen.**<br>**3. User presses on the specific diet plan that he/she wants to view.**<br>**4. The application displays the diet plan information.**<br>**5. The user can navigate back to the "Generate diet plan" screen to generate a new personalized diet plan.** | | |
| Sub-flows**:-** | | |
| Alternate/Exceptional Flows: - | | |

| Use Case Name**: Record progress** | ID**:** **UC004** | Importance Level**:** **Moderate** |
|---|---|---|
| Primary Actor**: User** | Use Case Type**: Detail, Real** | |
| Stakeholders and Interests**:**<br><br>**User – wants to record his/her daily progress to keep track of the body condition in daily basis.** | | |
| Brief Description**: The use case describes how the user can record his/her body condition in daily basis.** | | |
| Trigger**: User wants to record his/her body condition in daily bas** | | |
| Relationships:<br>Association **: User**<br>Include **: View progress**<br>Extend **: -**<br>Generalization**: -** | | |
| Normal Flow of Events**:**<br>**1. The user navigates to the "Record Progress" screen.**<br>**2. The application will display a form for the user to input their progress information such as weight or measurement manually.**<br>**3. The application validates the user's input and stores the information in the user's profile.**<br>**4. The application provides feedback to the user on their progress by indicating the level of user's health.**<br>**5. The user can choose to proceed to view their other health attribute's progress history or proceed to other screen.** | | |
| Sub-flows**: -** | | |
| Alternate/Exceptional Flows: - | | |

| Use Case Name: **View Progress** | ID: **UC005** | Importance Level: **Moderate** |
|---|---|---|
| Primary Actor: **User** | Use Case Type: **Detail, Real** | |
| Stakeholders and Interests:<br><br>**User – wants to view the visual representations of progress such as chart and graph.** | | |
| Brief Description: **The use case describes how the user can view the visual representations of progress such as chart and graph.** | | |
| Trigger: **User wants to view the visual representations of progress such as chart and graph.** | | |
| Relationships:<br>Association : **User**<br>Include : **-**<br>Extend : **-**<br>Generalization: **-** | | |
| Normal Flow of Events:<br>1. **User wishes to view his current or past progress data.**<br>2. **The user navigates to the "View Progress" screen in the application.**<br>3. **The application displays a list of user's previously recorded progress data, such as BMR, BMI or body fat percentage in a chart format.**<br>4. **The user can also choose to filter the progress data accordingly using data range, measurement type or other criteria.**<br>5. **User can choose to redirect back other screen after viewing the progress chart.** | | |
| Sub-flows: **-** | | |
| Alternate/Exceptional Flows:- | | |

| Use Case Name**: Use chatbot** | ID**:** **UC008** | Importance Level**: High** |
|---|---|---|
| Primary Actor**: User** | Use Case Type**: Detail, Real** | |
| Stakeholders and Interests**:**<br><br>**User – wants to use chatbot feature that allows users to ask questions about their diet plan and receive personalized responses.** | | |
| Brief Description**: This use case describes how the user can use chatbot feature that allow users to ask questions about their diet plan and receive personalized responses.** | | |
| Trigger**: User wants to use chatbot feature that allows users to ask questions about their diet plan and receive personalized responses.** | | |
| Relationships:<br>Association **: User**<br>Include **: -**<br>Extend **: -**<br>Generalization**: -** | | |
| Normal Flow of Events**:**<br>1. **The user selects on the "Chatbot" button in the home screen.**<br>2. **The application displays the chatbot interface.**<br>3. **The user input a message or questions for the chatbot.**<br>4. **The chatbot uses the AI algorithm to analyze the message and generate a response.**<br>5. **The user can continue the conversation with the chatbot by sending additional messages.**<br>6. **The chatbot can provide personalized diet advice and tips based on the user's profile information and progress.**<br>7. **The user can end the conversation by selecting the "End Chat" button.**<br>8. **The application will store the conversation history for user's future reference.** | | |
| Sub-flows**: -** | | |
| Alternate/Exceptional Flows: - | | |

| Use Case Name: **Receive feedback** | ID: **UC009** | Importance Level: **High** |
|---|---|---|
| Primary Actor: **User** | Use Case Type: **Detail, Real** | |
| Stakeholders and Interests:<br><br>**User – wants to provide feedback**<br>**Project Team – The team will improvise the application by updating the application.** | | |
| Brief Description: **This use case describes how a user provides a feedback and how the project team refer to the feedback and improve the application,** | | |
| Trigger: **The user discovers some error or dissatisfaction of the application and provide feedback to the project team, the project team receives and improve the application based on the feedback.** | | |
| Relationships:<br>Association : **User, Project Team**<br>Include : **-**<br>Extend : **-**<br>Generalization: **-** | | |
| Normal Flow of Events:<br>1. **The user discovers some error or dissatisfaction of the application and provide feedback to project team through the application.**<br>2. **The project team will view the feedback and analyze the problem.**<br>3. **The project team will reply to the feedback given by the user and the user will view the feedback.**<br>4. **The project team will make a report and analyze on the application problem.**<br>5. **The project team will improvise the application based on the problem by updating the application.** | | |
| Sub-flows:**-** | | |
| Alternate/Exceptional Flows:- | | |

## 4.5 Interface Flow Diagram

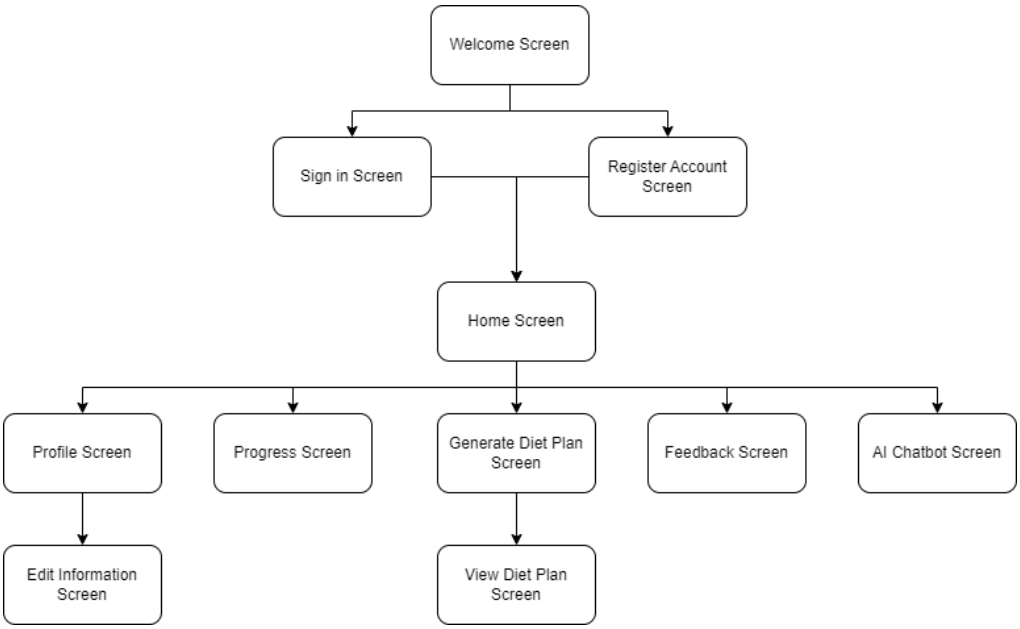



Figure 4.22: Interface flow diagram of the proposed system

## 4.6 Prototype

### 4.6.1 Mobile Application Interface Design

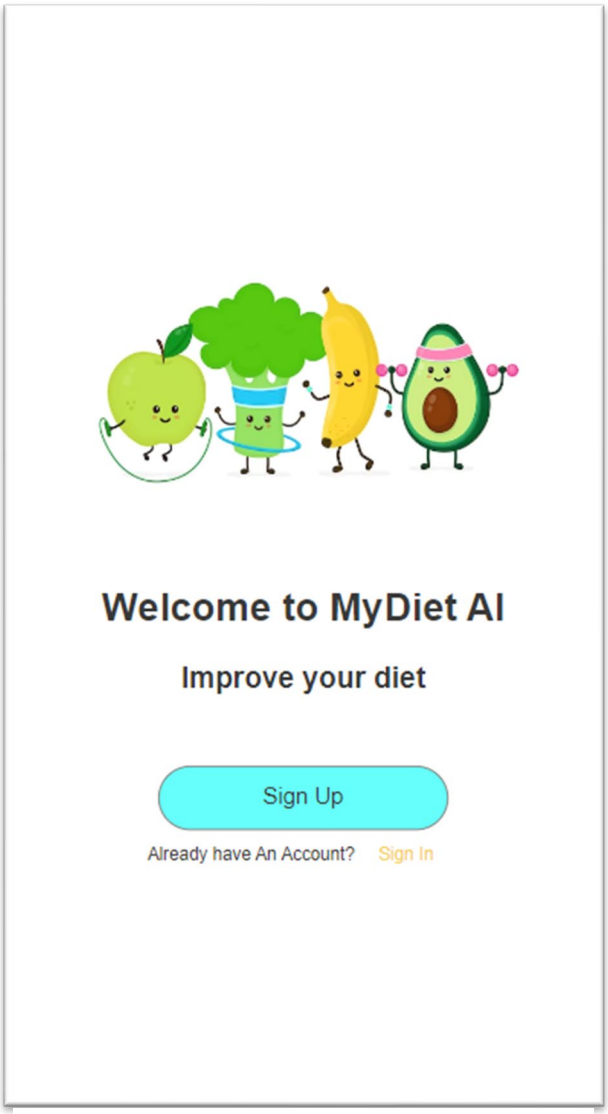



Figure 4.23: Welcome Screen.

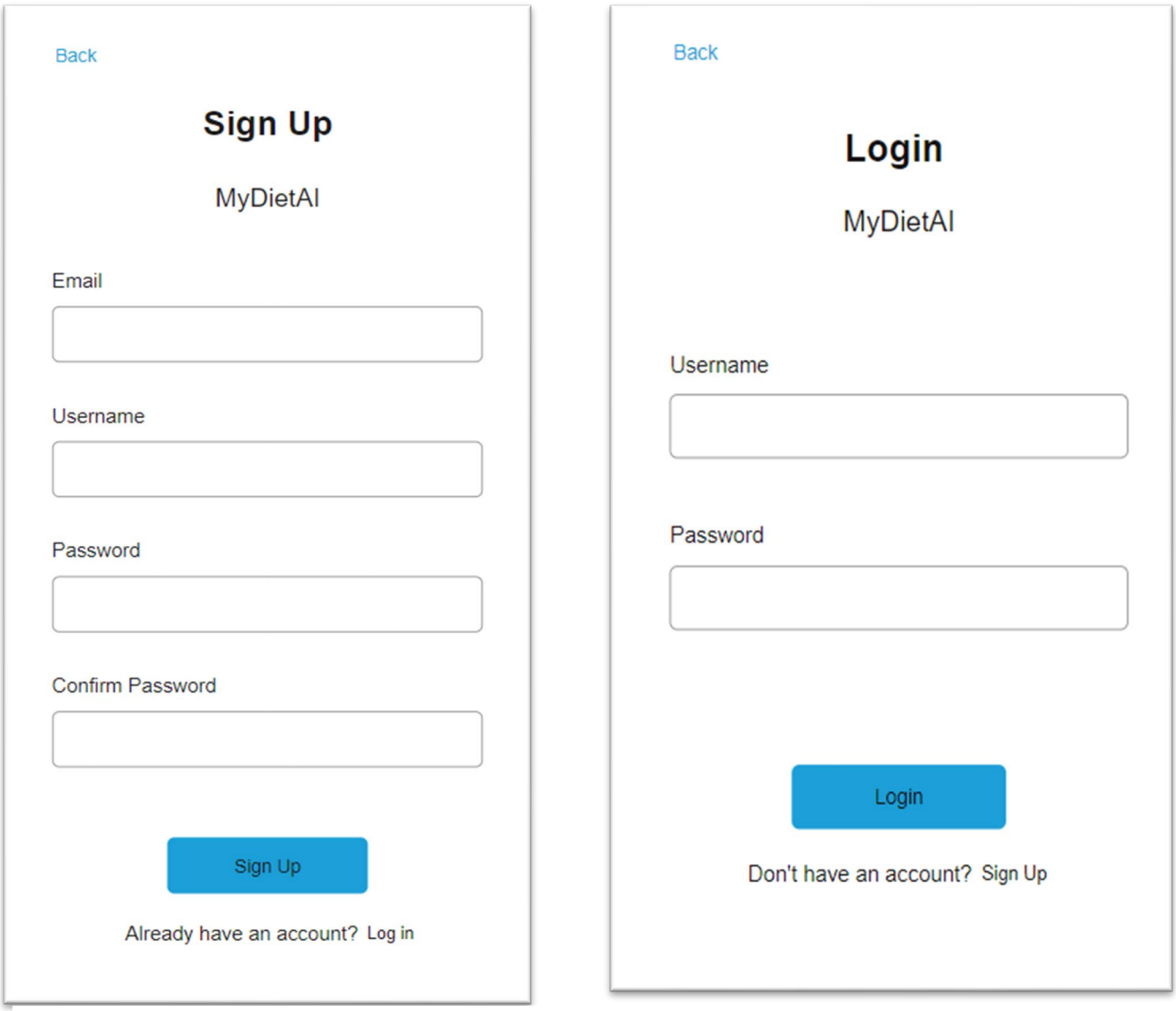



Figure 4.24: Sign Up and Login Screen.

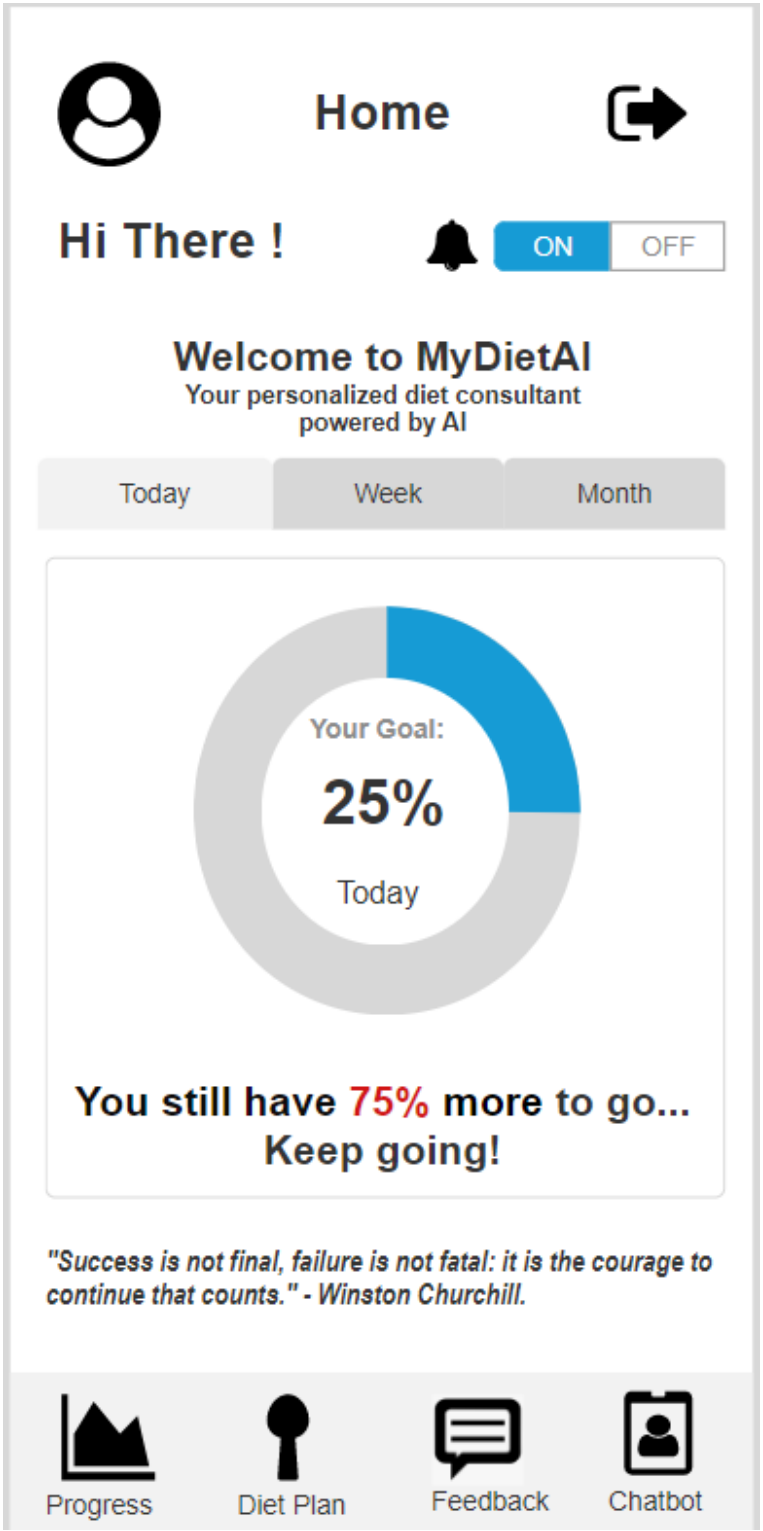



Figure 4.25: Home Screen.

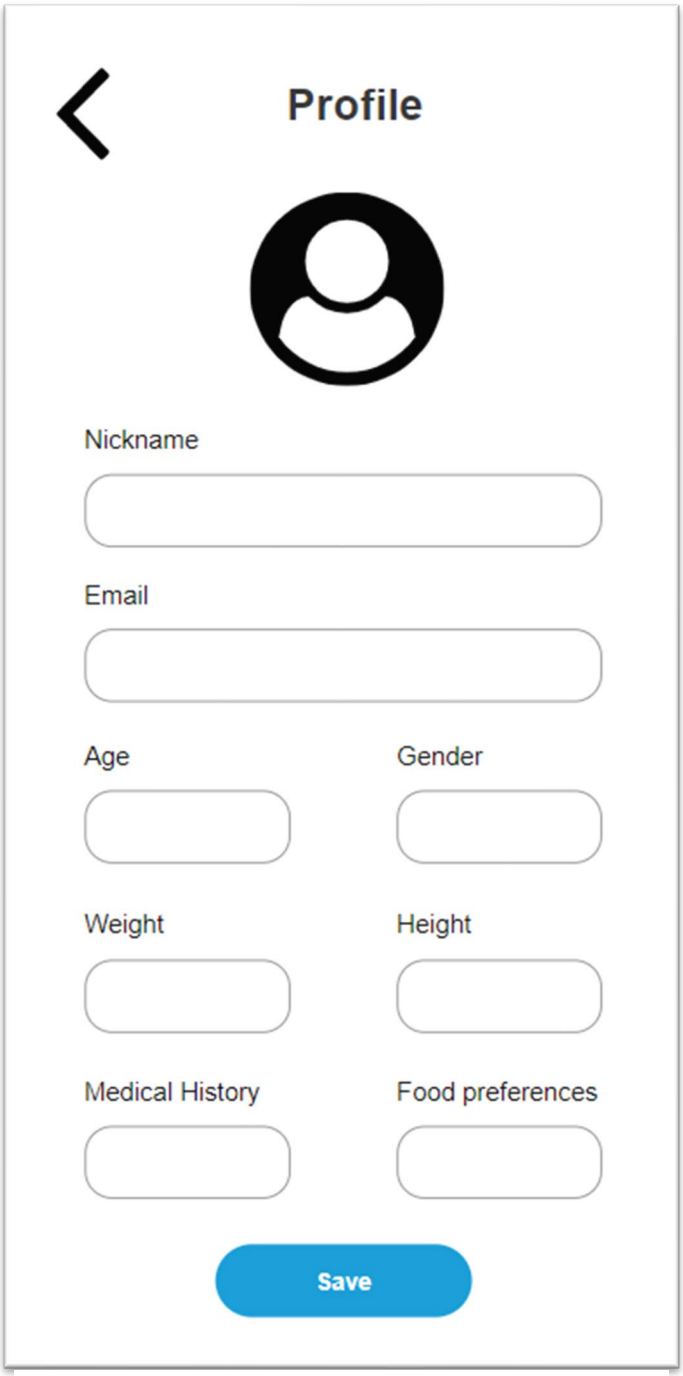



Figure 4.26: Profile Screen.

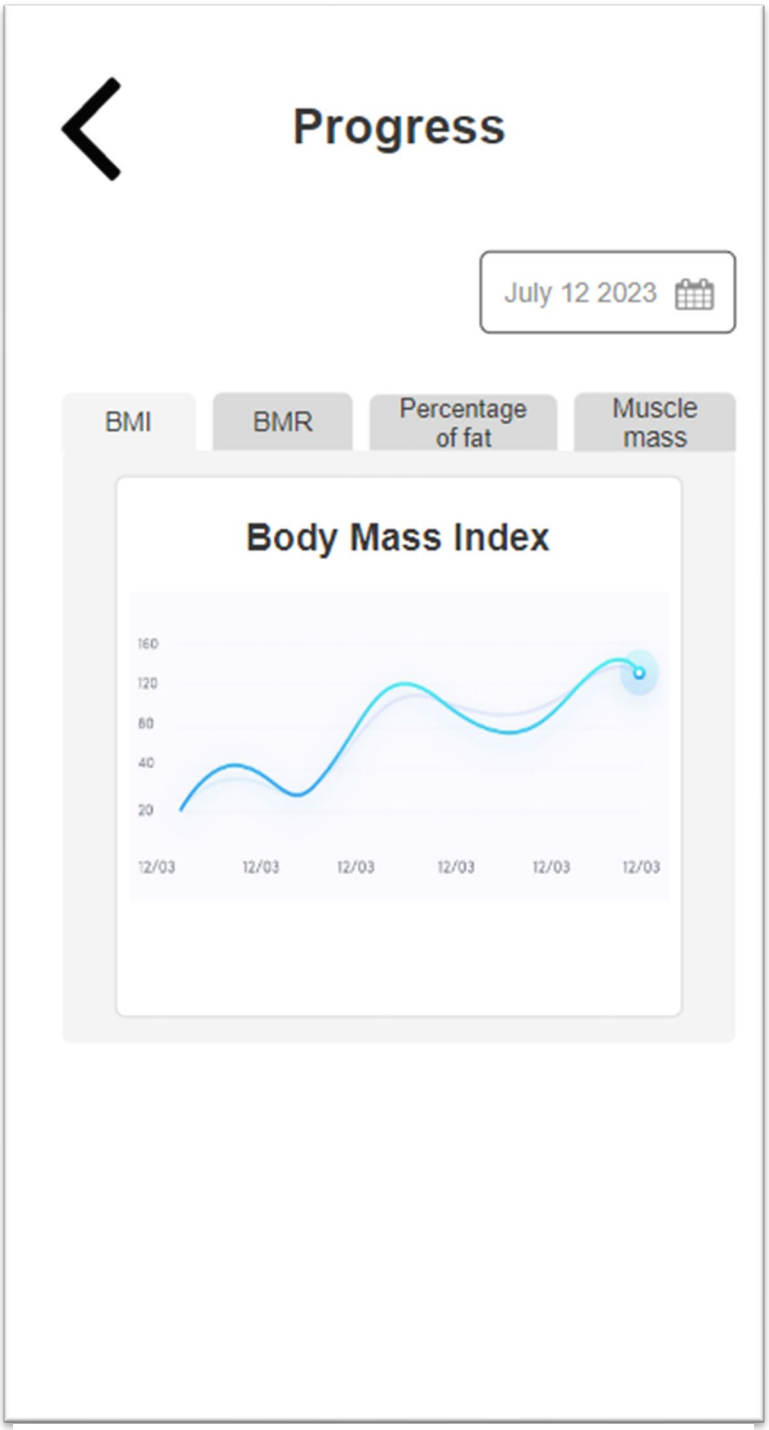



Figure 4.27: Progress Screen.

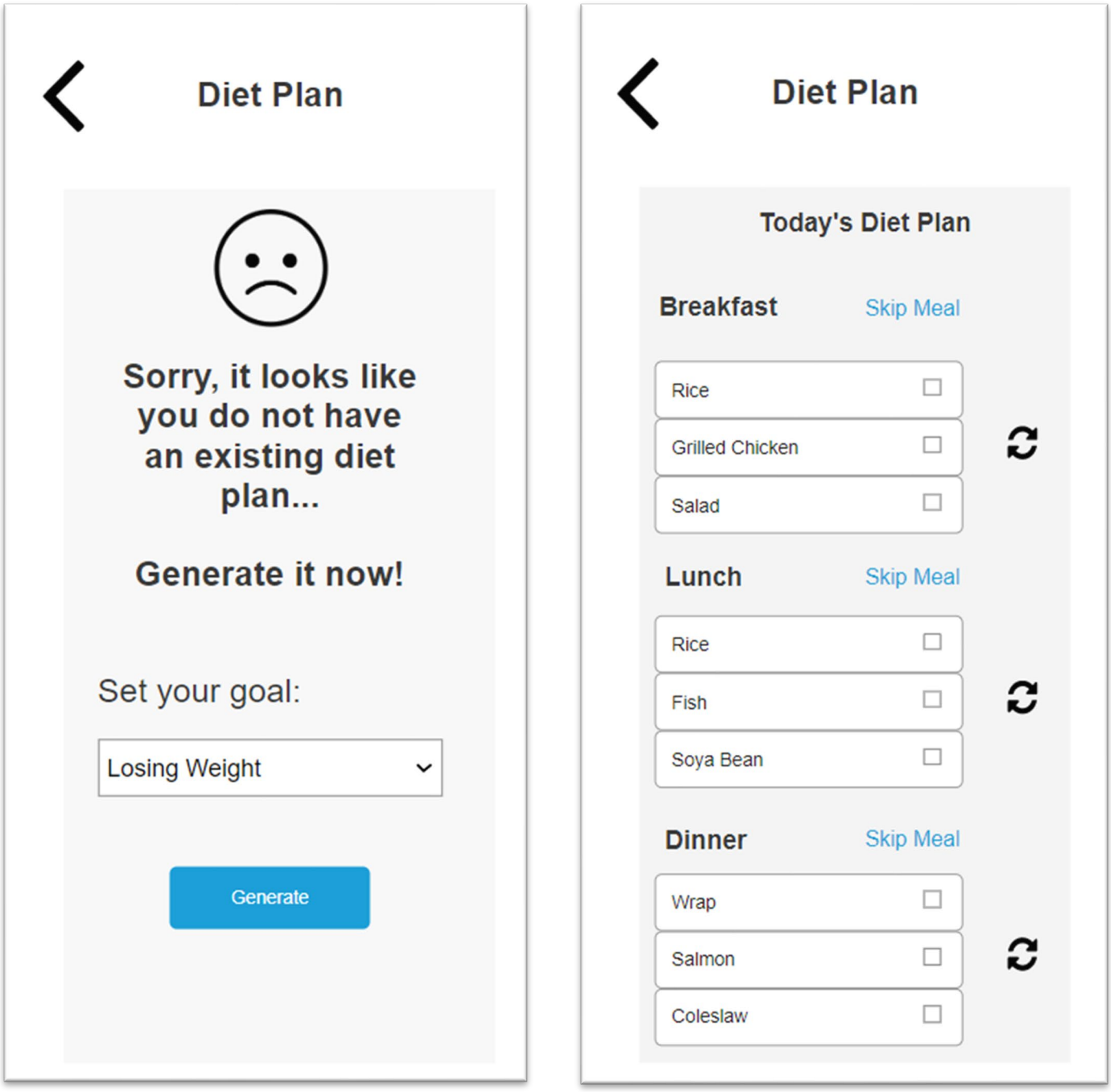



Figure 4.28: Diet Plan Screen

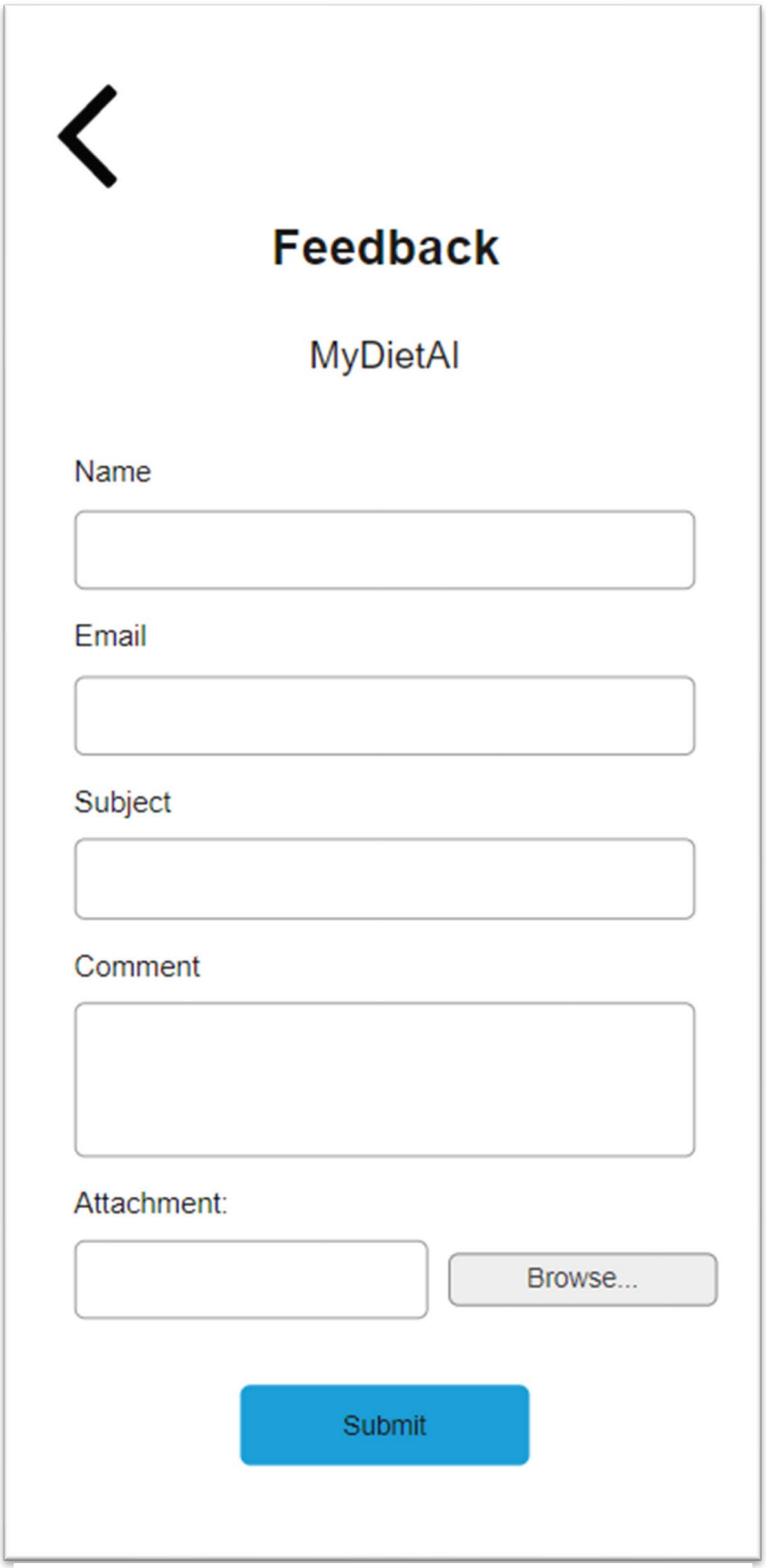



Figure 4.29: Feedback Screen.

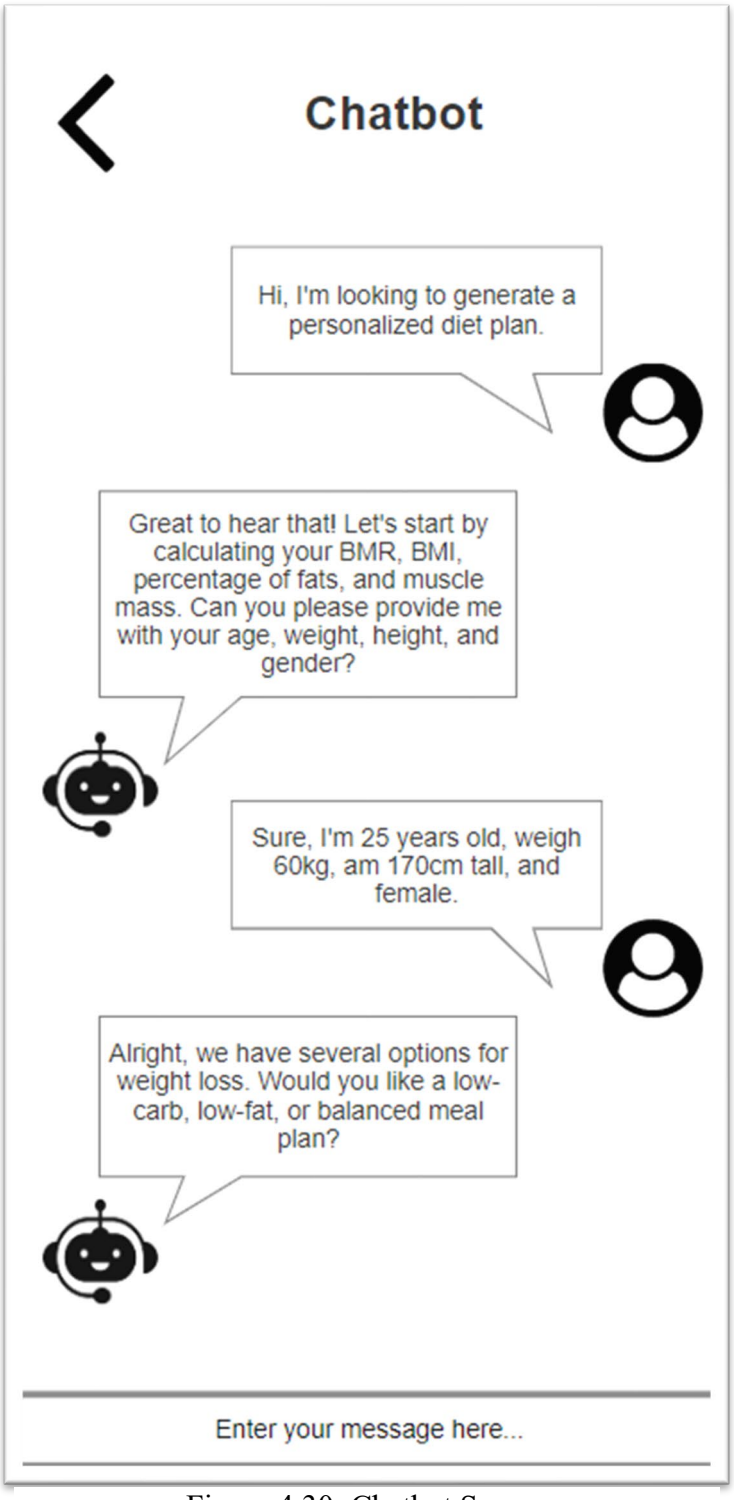



Figure 4.30: Chatbot Screen

# CHAPTER 5

# DESIGN

## 5.1 Introduction

This chapter showcases the prototype designs, modelling diagrams, and system architecture of the implemented system. The system architectural design is presented to illustrate the interconnections and communication among the implemented system, backend server, database, and external APIs. Modelling diagrams, including entity relationship diagrams (ERD), a data dictionary, and activity diagrams, have been generated to elucidate the structure of database, flow, and operational logic of the implemented system. Furthermore, a high-fidelity prototype design has been developed as a reference for the system's user interface implementation, with documented details for comparison and analysis.

## 5.2 System Architecture Design

The system architecture design is depicted in Figure 5.1. It illustrates the implemented system's behaviour and structure of operation.

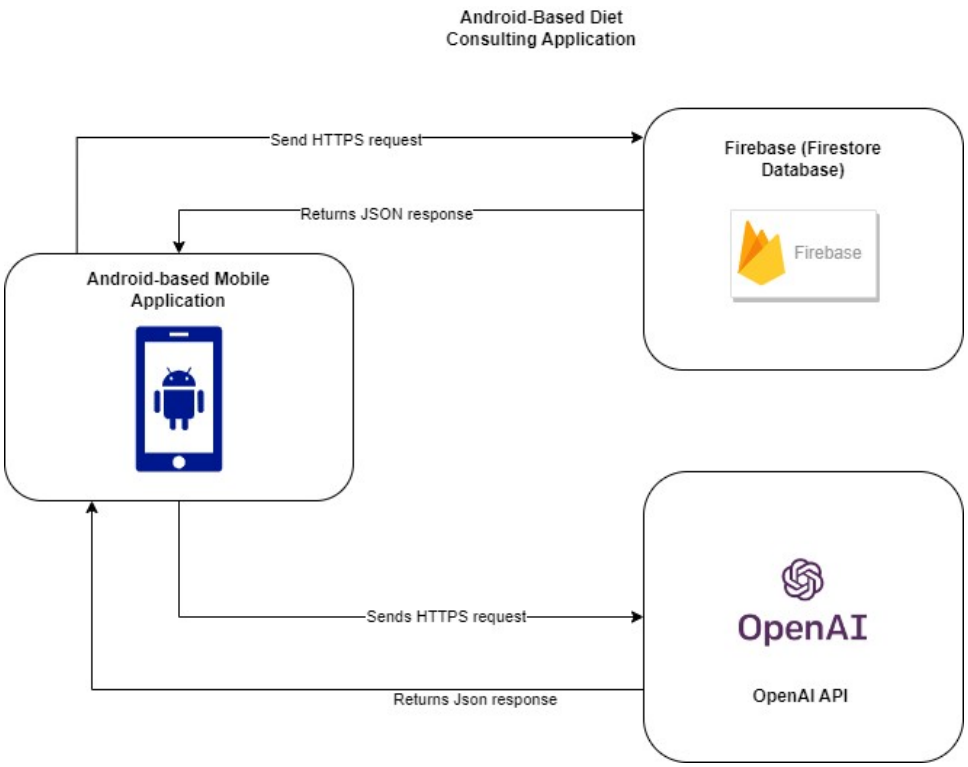



Figure 5.1: System Architecture Design

The implemented system is an Android-based mobile application developed using the Flutter framework. This application interacts seamlessly with Firebase's Firestore database for storing crucial user data, including user progress, diet plans, and feedback. Firebase's authentication system is employed for user login and signup procedures, ensuring a secure and user-friendly experience.

Additionally, the implemented system integrates with the OpenAI API to access advanced AI-driven features. Through this API, the application can provide personalized meal planning and engage in virtual diet chatbot interactions. Users can receive tailored advice and information related to their goals, food preferences, etc, all within the app's interface.

Furthermore, to enhance the application's functionality, we've leveraged external APIs, such as the OpenAI API, to retrieve and provide users with the necessary information for specific features and functionalities. This streamlined approach ensures that the app efficiently delivers valuable content and services to users.

The overall architecture of the implemented system fosters smooth communication between the mobile application, Firebase's Firestore database, and the OpenAI API. When users interact with the app, it securely sends and retrieves data from Firebase's Firestore database, offering a seamless and responsive user experience.

## 5.3 Data Model Design

The modelling diagrams produced encompass various elements, including data models, a data dictionary, and activity diagrams. The initial step involved the creation of Entity Relationship Diagram (ERD), which elucidated how the functional requirements influenced the interconnections among entities. Additionally, the creation of activity diagrams offered a visual representation of various possible activities.

### 5.3.1 System Database Design

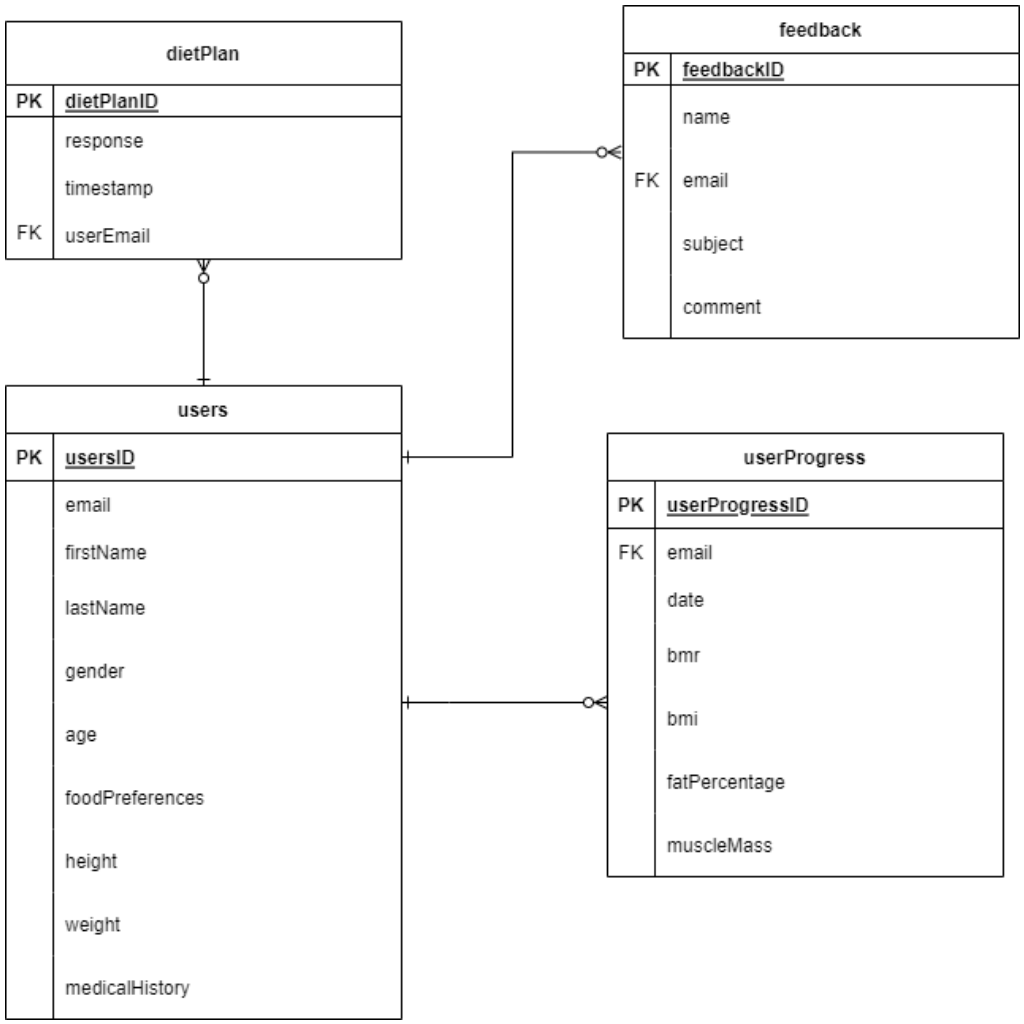



Figure 5.2: ERD Diagram of Android-based diet consultant application

### 5.3.2 Data Dictionary

**Table Name: users**

Table 5.1: Data dictionary for users table.

| Column Name | Description | Data Type | Size | PK/FK | Nullable |
|---|---|---|---|---|---|
| usersID | Unique identification of user | SERIAL | - | PK | No |
| email | User's email address | VARCHAR | 255 | FK | No |
| firstName | User's first name | VARCHAR | 25 | - | No |
| lastName | User's last name | VARCHAR | 25 | - | No |
| gender | User's gender | VARCHAR | 6 | - | No |
| age | User's age | NUMERIC | 3,2 | - | No |
| foodPreferences | User's food preferences | VARCHAR | 25 | - | Yes |
| height | User's height | NUMERIC | 3,2 | - | No |
| weight | User's weight | NUMERIC | 3,2 | - | No |
| medicalHistory | User's medical history | VARCHAR | 25 | - | Yes |

**Table Name: dietPlan**

Table 5.2 : Data dictionary for dietPlan table.

| Column Name | Description | Data Type | Size | PK/FK | Nullable |
|---|---|---|---|---|---|
| dietPlanID | Unique identification of diet plan | SERIAL | - | PK | No |
| response | Diet plan's response | VARCHAR | 255 | - | No |
| Timestamp | Diet plan's timestamp | VARCHAR | 50 | - | No |
| userEmail | User's email address | VARCHAR | 255 | FK | No |

**Table Name: feedback**

Table 5.3 : Data dictionary for feedback table.

| Column Name | Description | Data Type | Size | PK/FK | Nullable |
|---|---|---|---|---|---|
| feedbackID | Unique identification of feedback | SERIAL | - | PK | No |
| name | User's name | VARCHAR | 50 | - | No |
| email | User's email | VARCHAR | 255 | FK | No |
| subject | User's subject | VARCHAR | 50 | - | No |
| comment | User's comment | VARCHAR | 255 | - | No |

**Table Name: userProgress**

Table 5.4 : Data dictionary for userProgress table.

| Column Name | Description | Data Type | Size | PK/FK | Nullable |
|---|---|---|---|---|---|
| userProgressID | Unique identification of user progress | SERIAL | - | PK | No |
| email | User's email address | VARCHAR | 255 | FK | No |
| date | User progress's date | VARCHAR | 50 | - | No |
| bmr | User's bmr | NUMERIC | 5,2 | - | No |
| bmi | User's bmi | NUMERIC | 3,2 | - | No |
| fatPercentage | User's fat percentage | NUMERIC | 3,2 | - | No |
| muscleMass | User's muscle mass | NUMERIC | 3,2 | - | No |

### 5.3.3 Activity Diagram

The activity diagrams presented in this section depict the potential workflows and activities within the implemented system. These diagrams offer a visual representation that aids in understanding the application's logical flow and the interactions between users and the system.

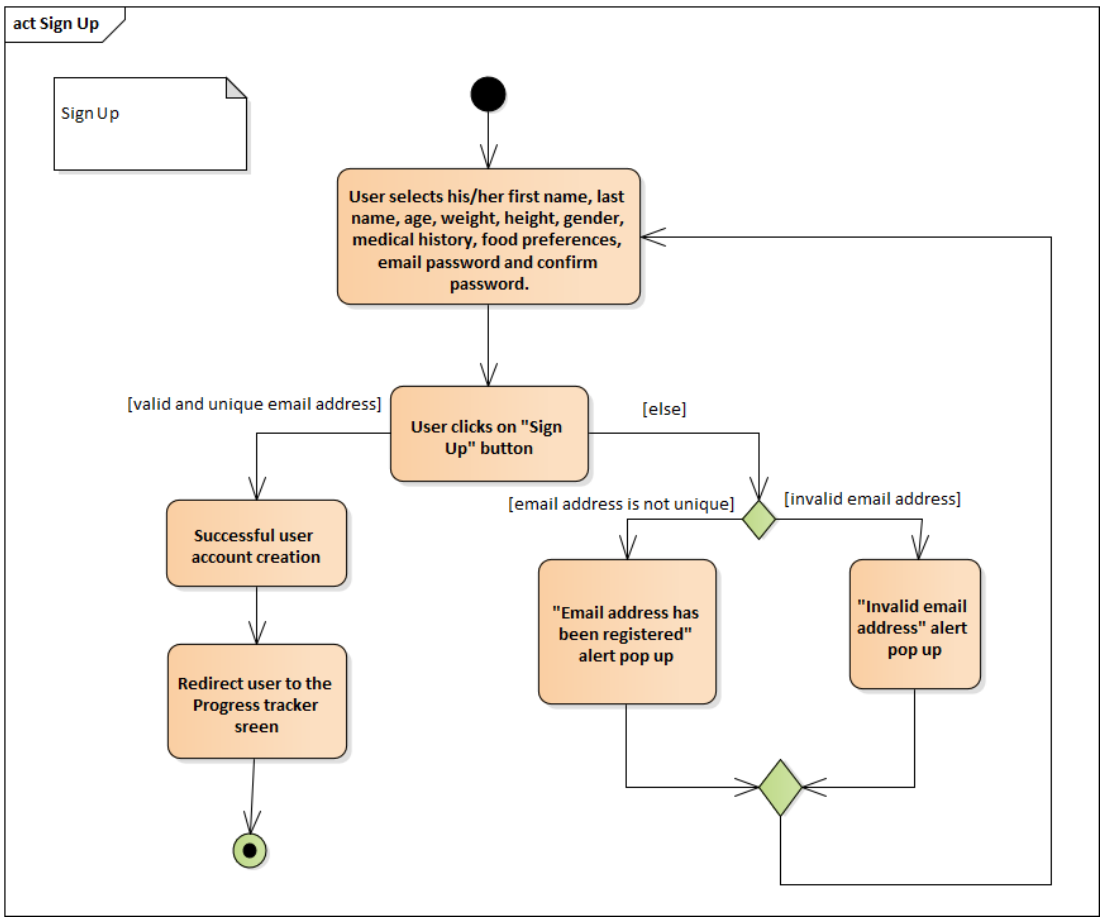



Figure 5.3 : Activity Diagram for Sign Up module.

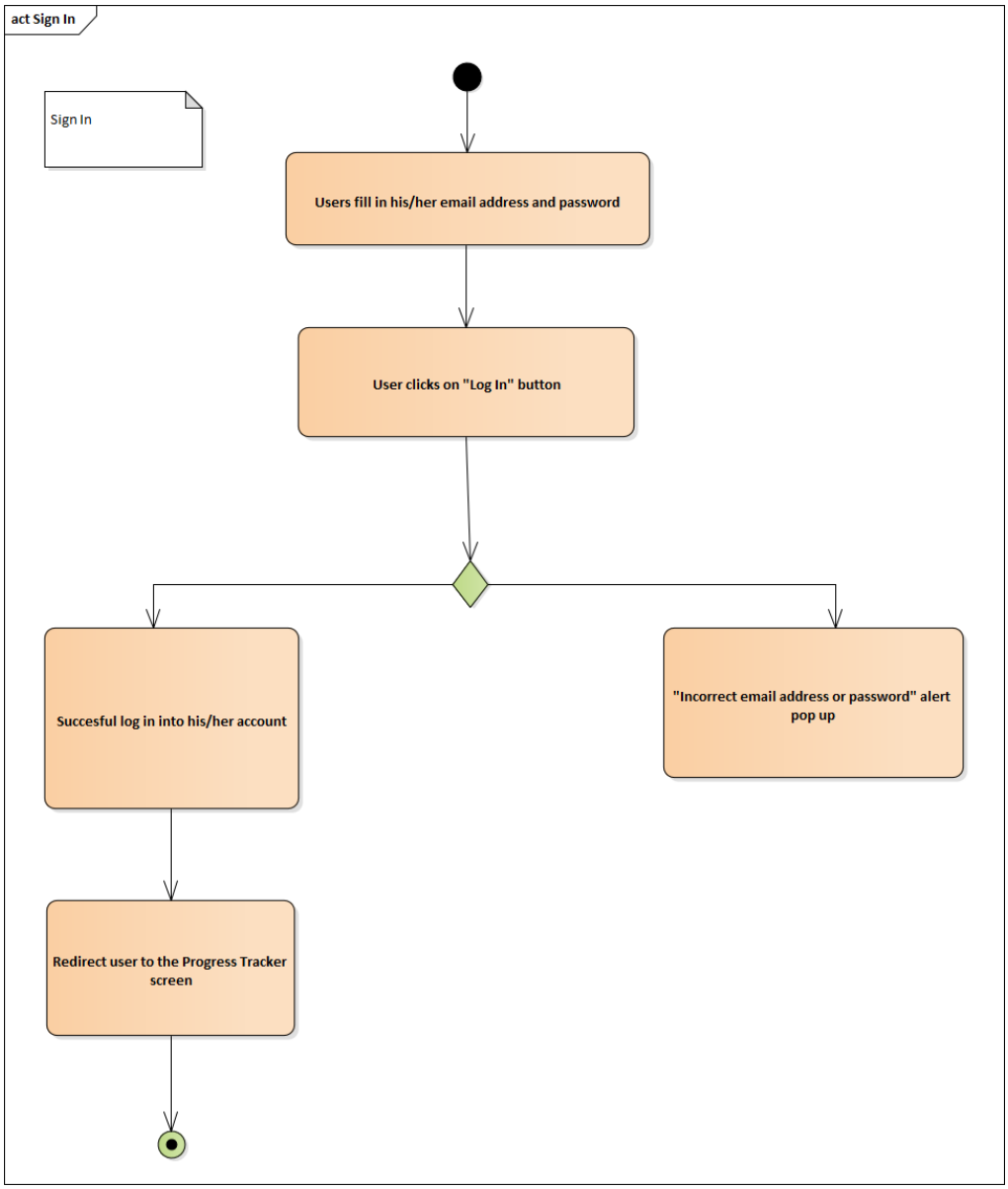



Figure 5.4 : Activity Diagram for Sign In module.

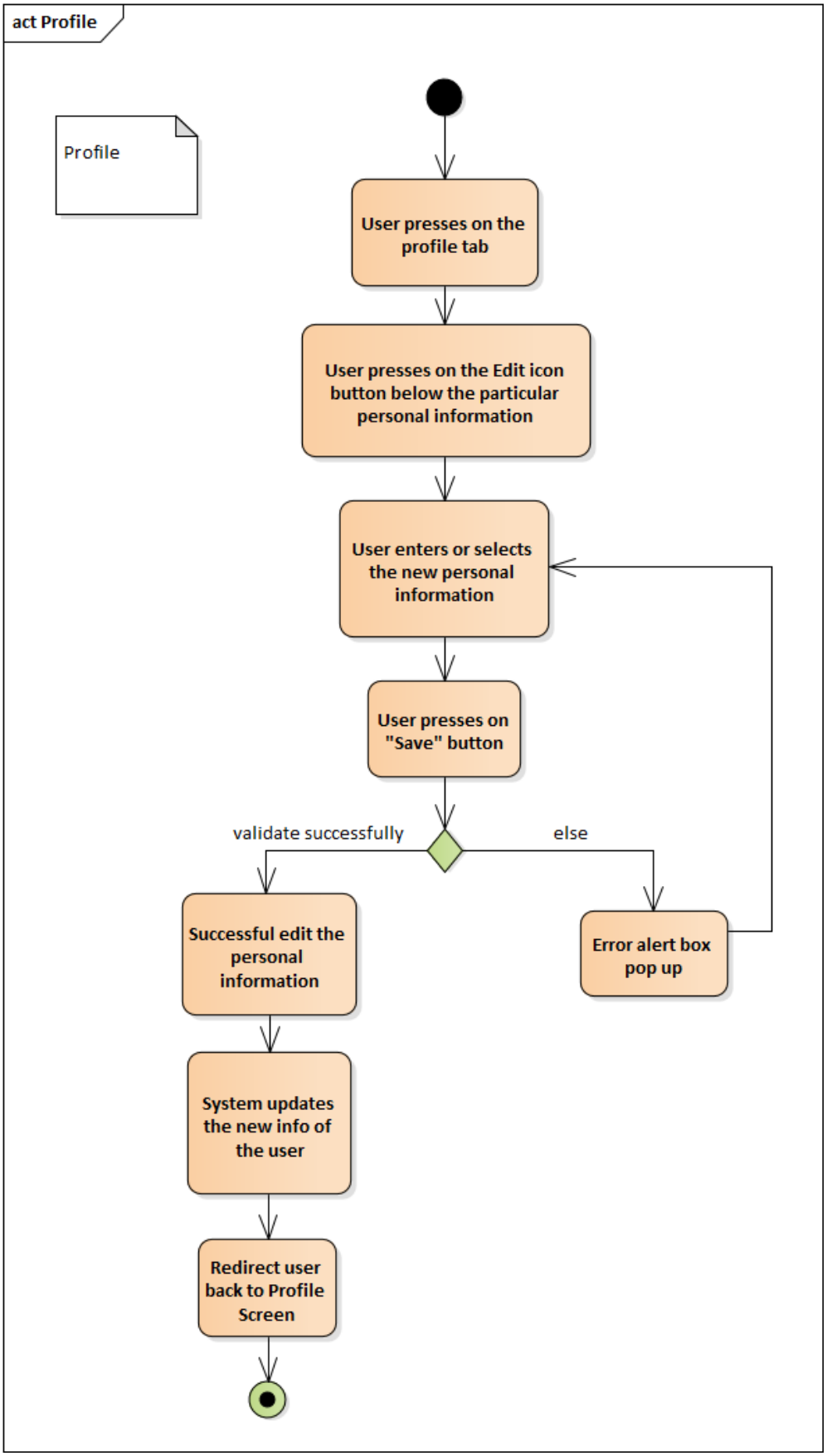



Figure 5.5 : Activity Diagram for Profile module.

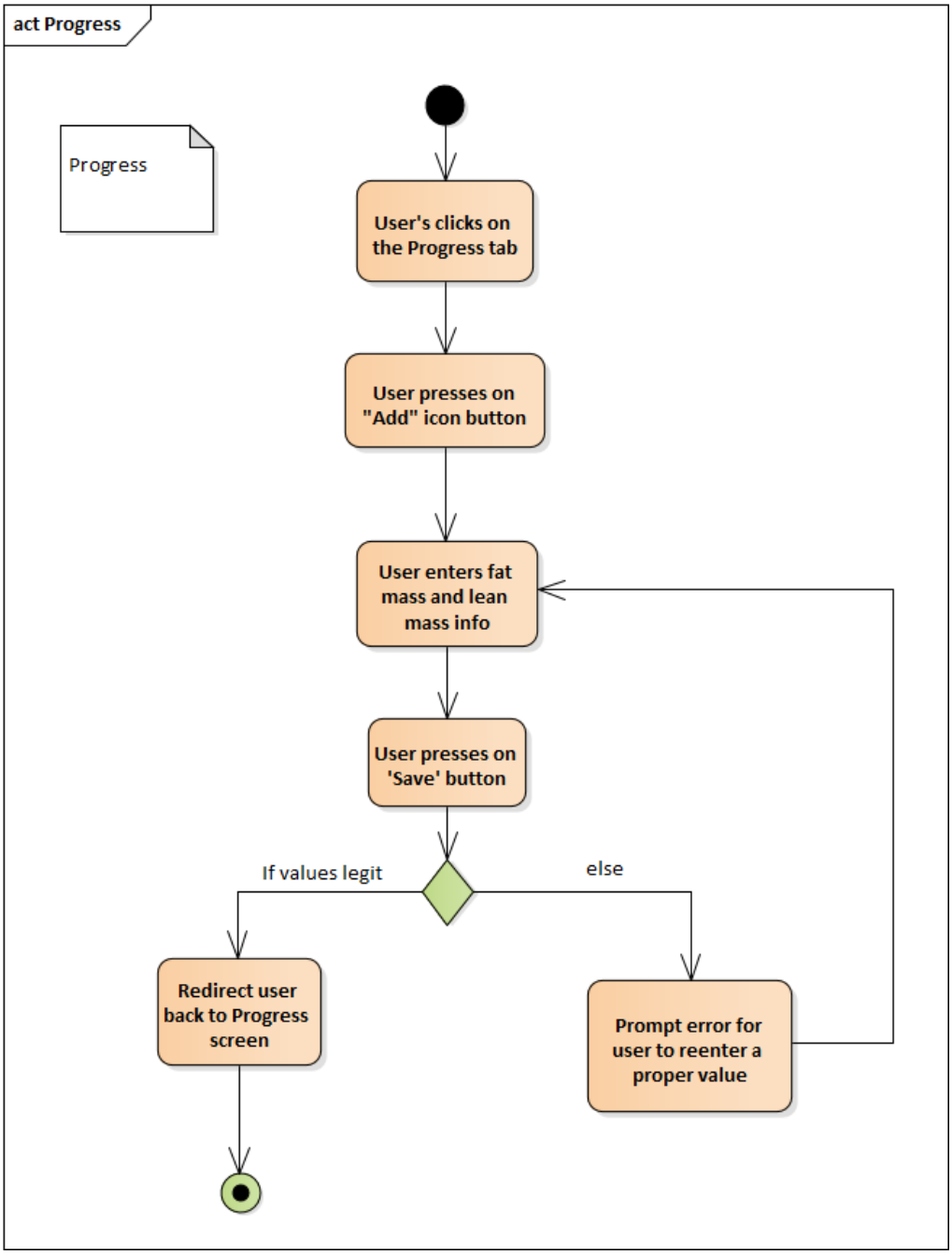



Figure 5.6 : Activity Diagram for Progress Tracker module

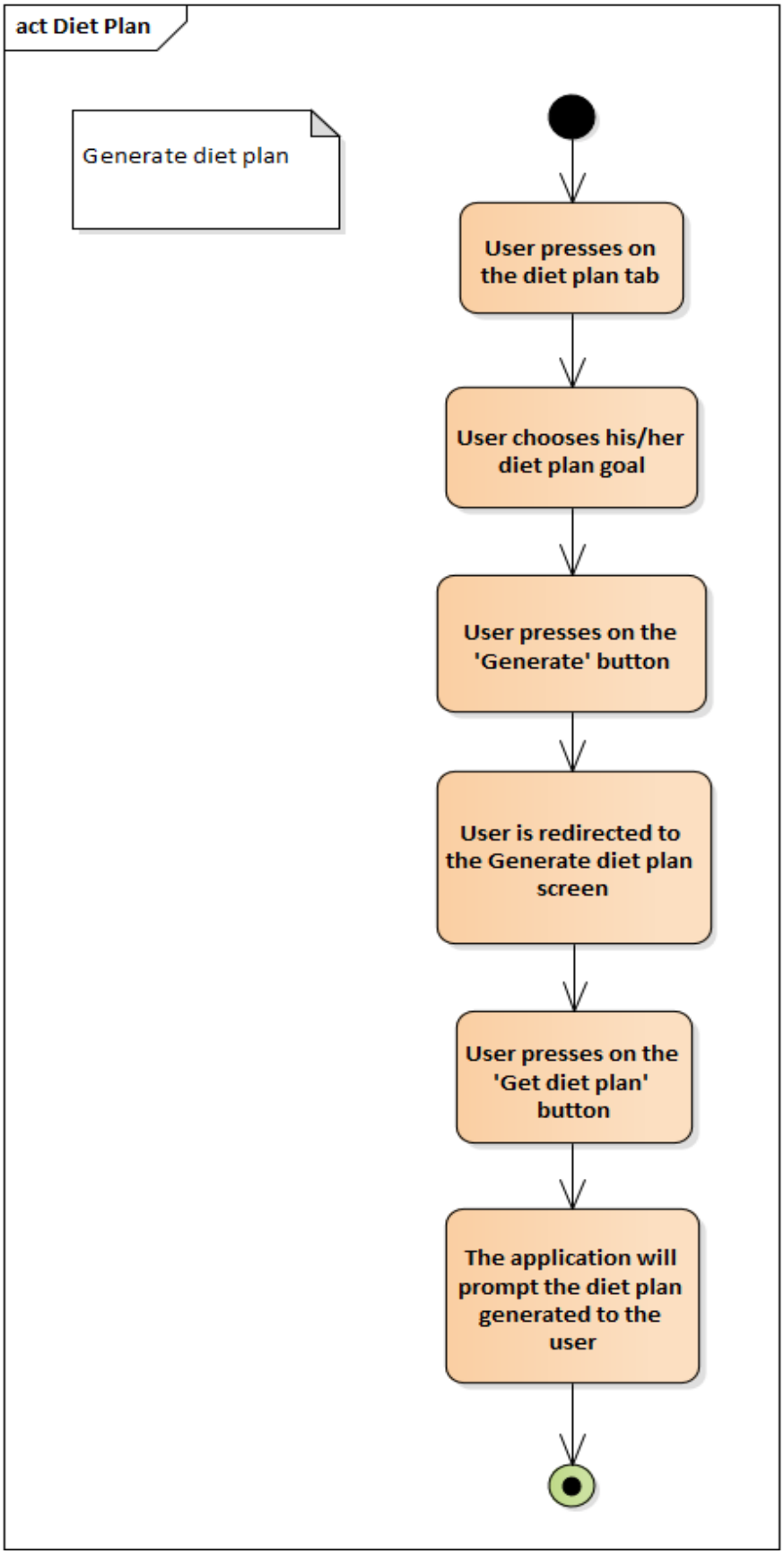



Figure 5.7 : Activity Diagram for Generate Diet Plan module.

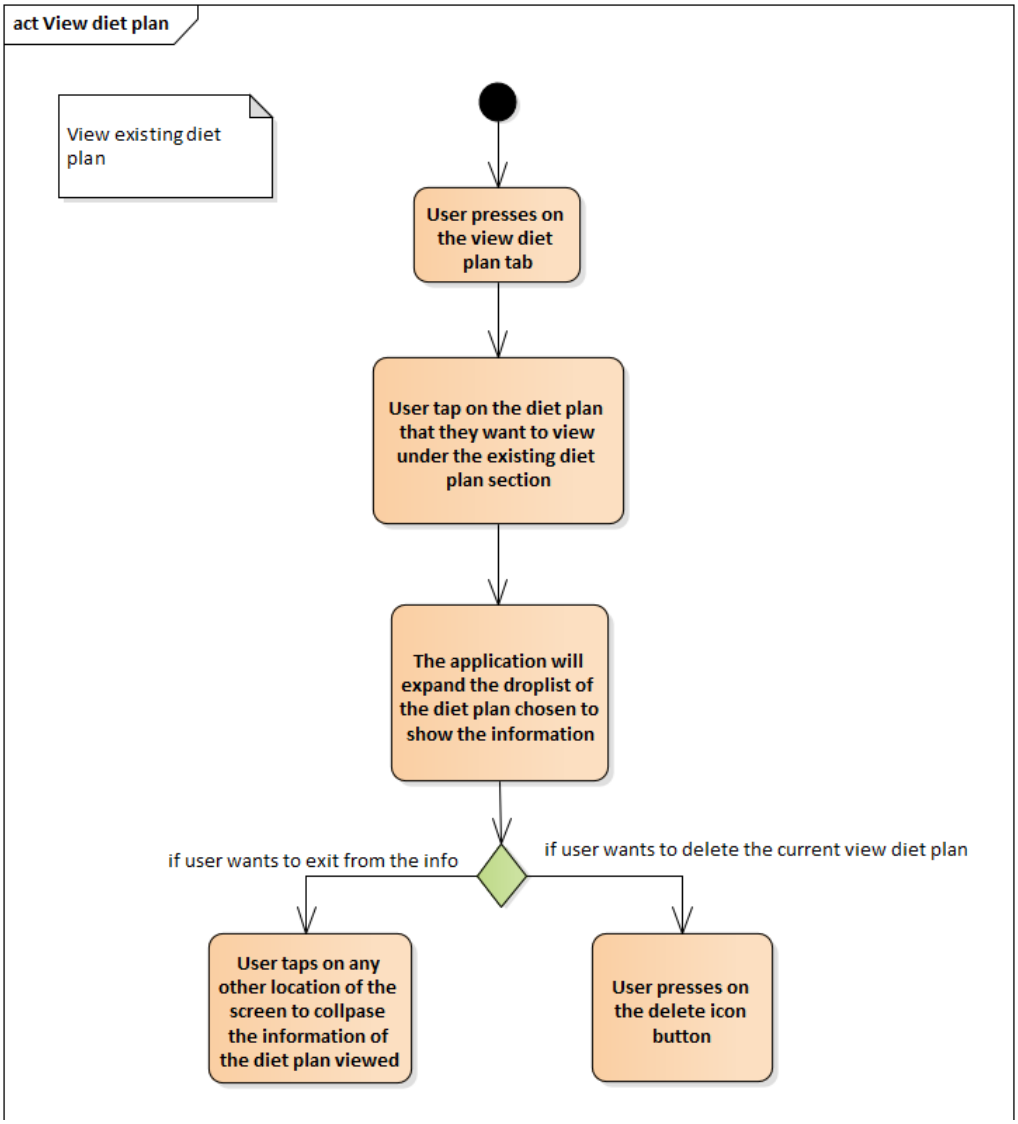



Figure 5.8 : Activity Diagram for View Existing Diet Plan module

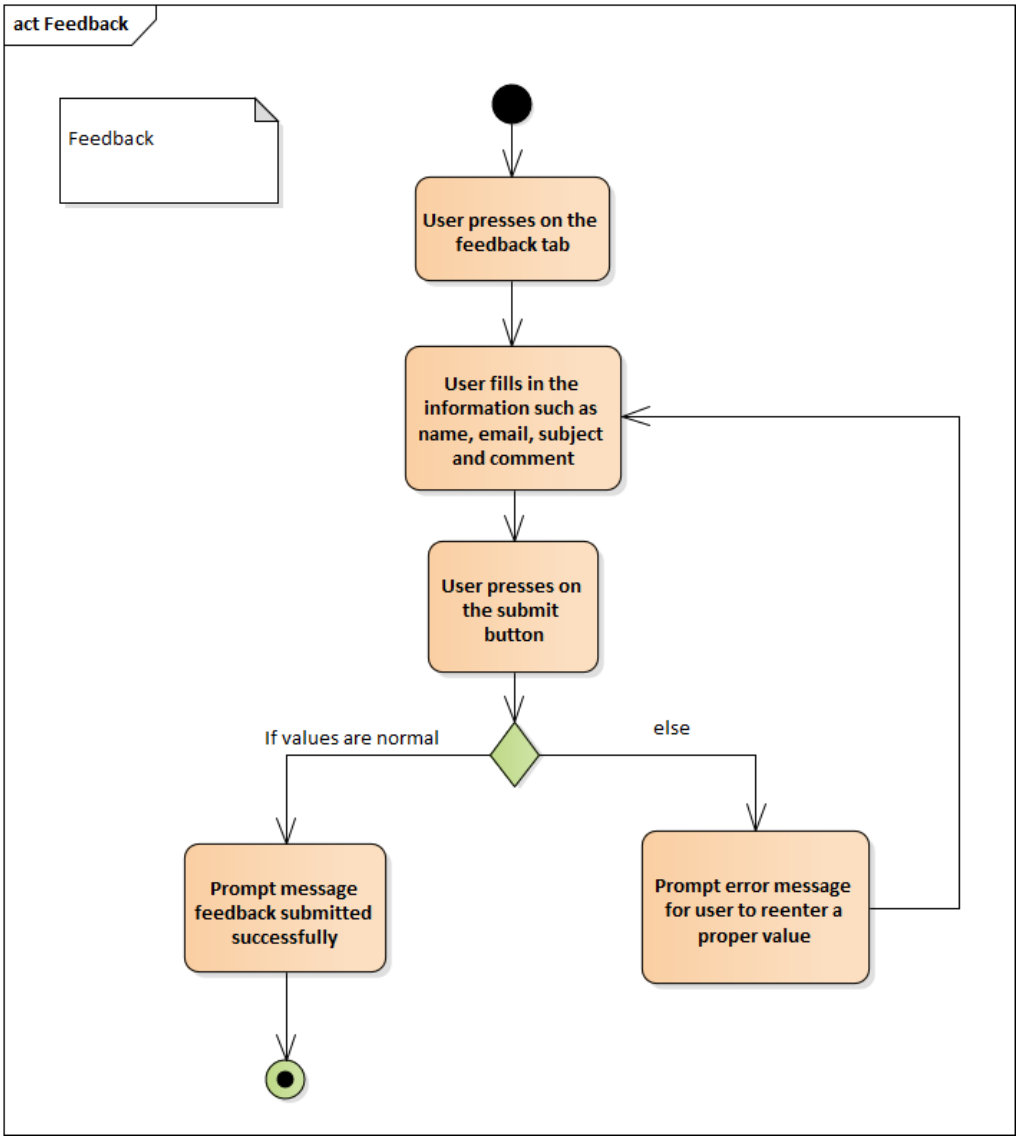



Figure 5.9 : Activity Diagram for Feedback module

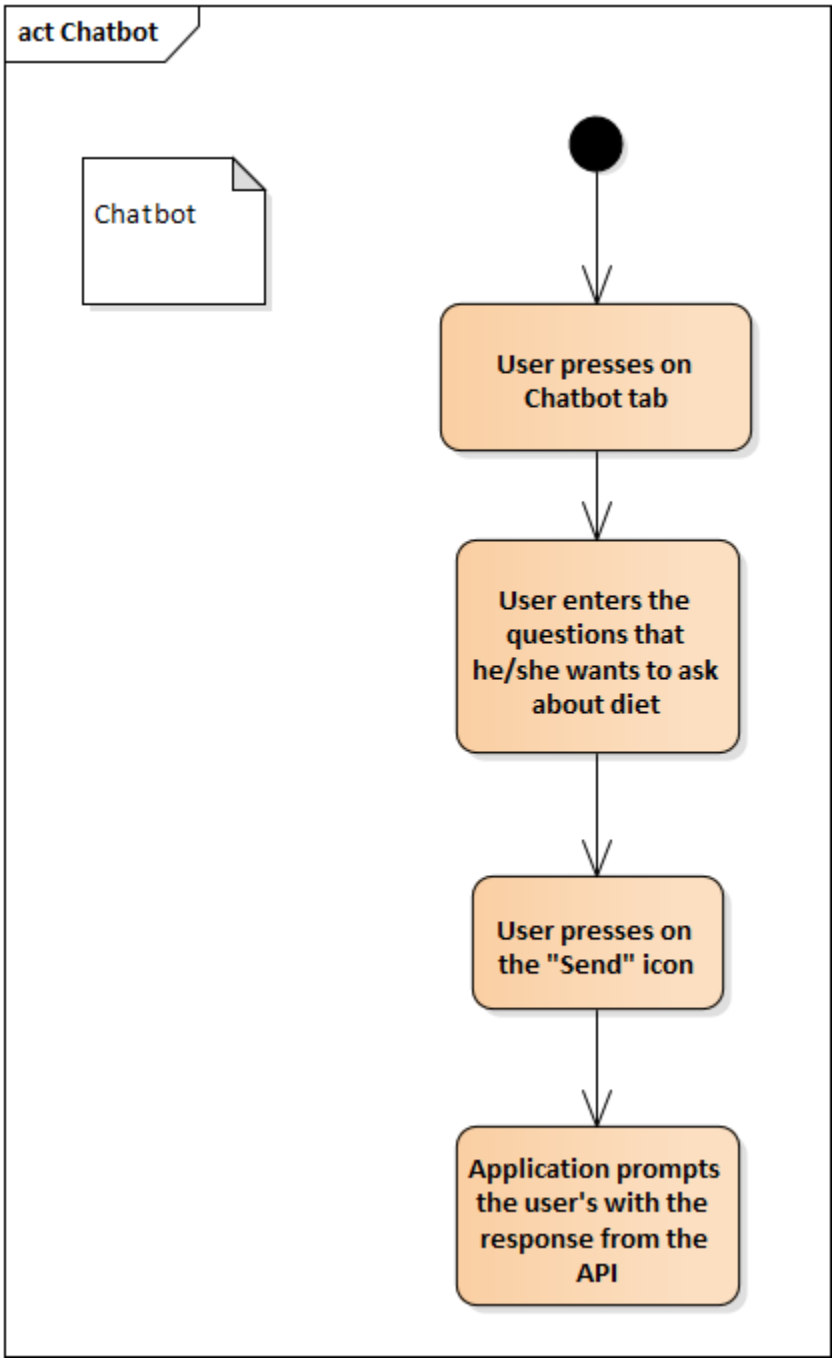



Figure 5.10 : Activity Diagram for Chatbot module

## 5.4 User Interface Designs

User interface design plays a pivotal role in the success of any software application as it serves as the bridge between the user and the application's functionality. A well-designed user interface enhances usability, making it intuitive and easy for users to navigate through the app. This is particularly important for this project, where users seek quick access to personalized diet plans and health-related information. When users find an app visually appealing and user-friendly, they are more likely to stay engaged, utilize the app regularly, and achieve their health goals effectively. These screenshots will serve as a testament to the importance placed on the simplicity of user interface, highlighting how it enhances usability, engagement, and trust for our users.

1. Onboarding Screen & Navigation UIs

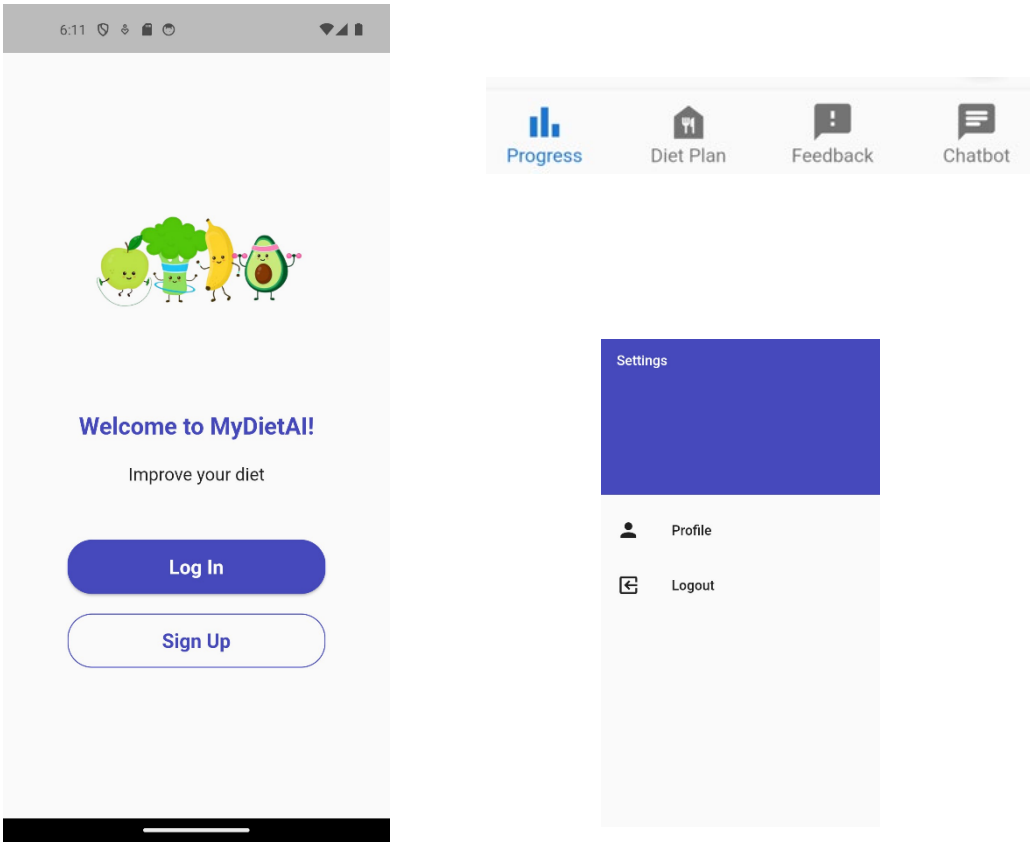



Figure 5.11: Actual Onboarding & Navigation UIs

2. Sign Up Screen

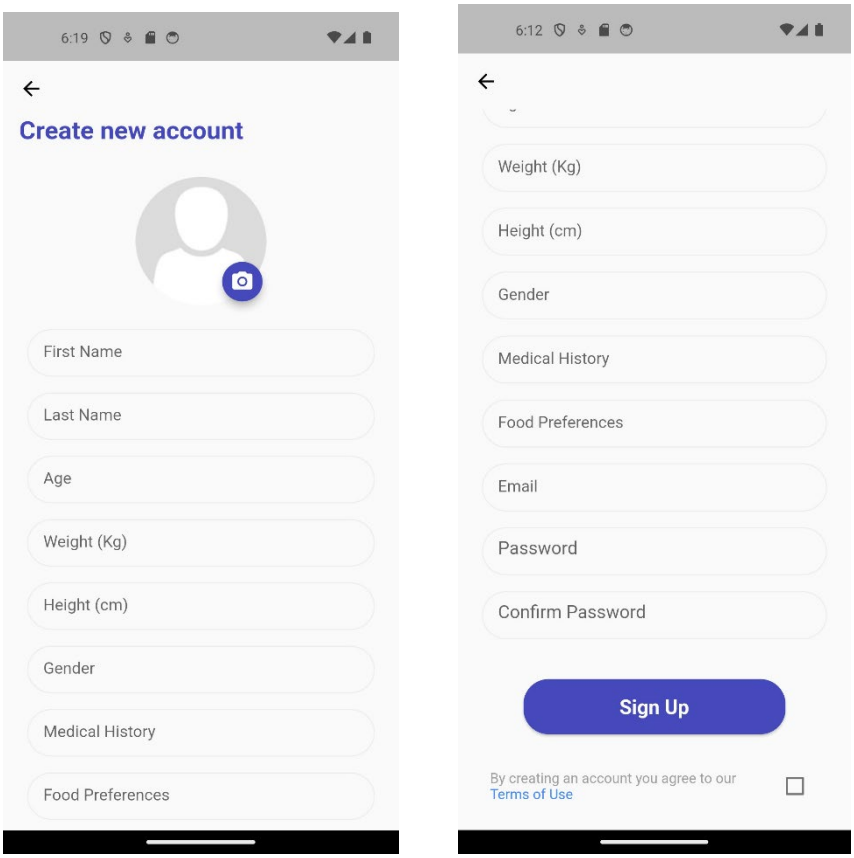



Figure 5.12 : Actual Sign Up Screen

3. Sign In Screen

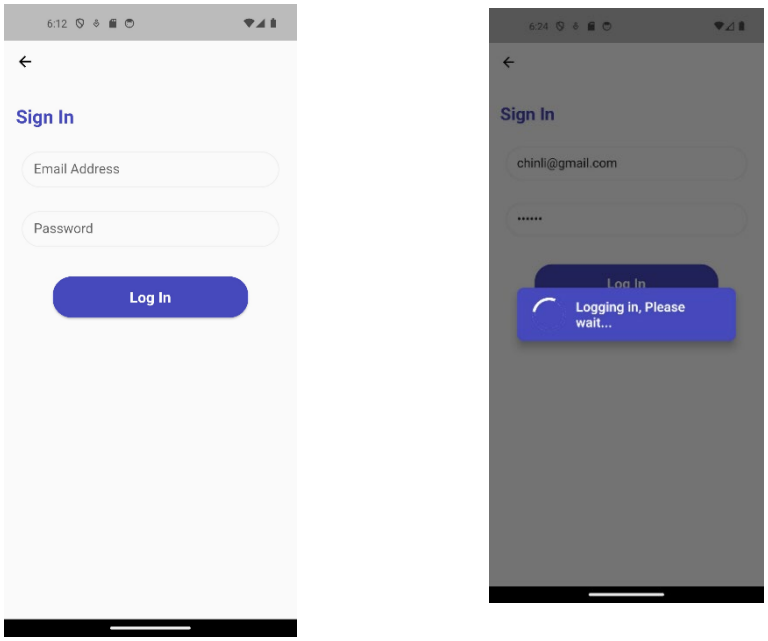



Figure 5.13 : Actual Sign In screen

4. Profile Screen

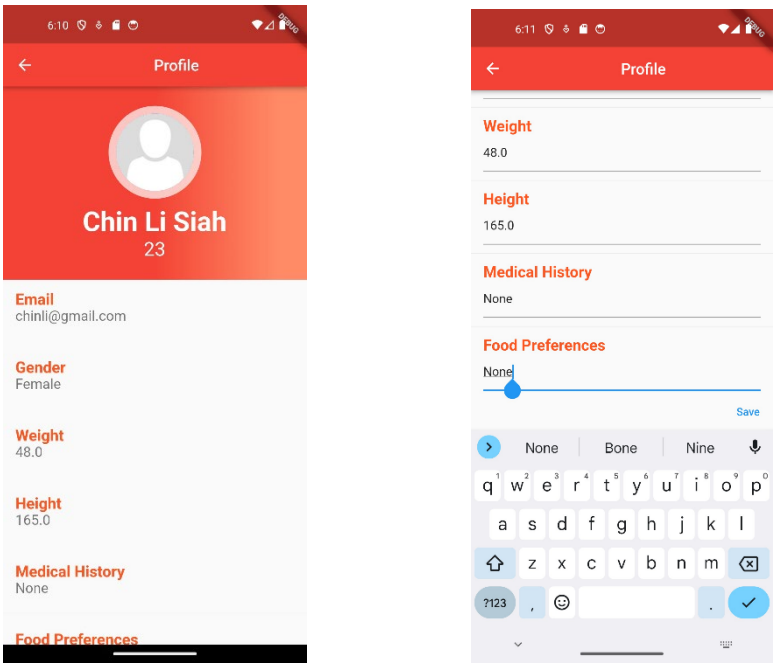



Figure 5.14 : Actual Profile Screen

5. Progress Screen

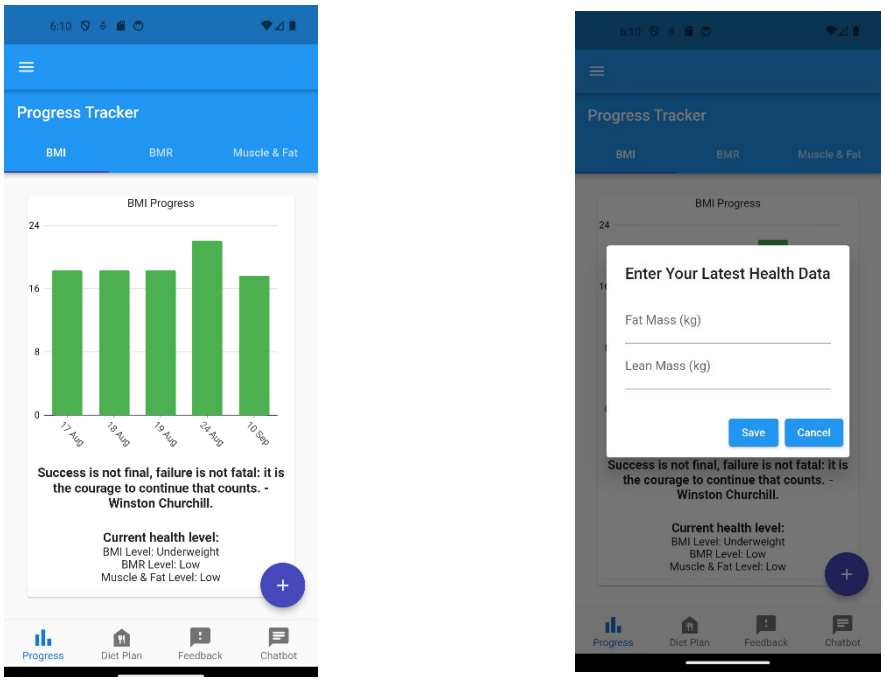
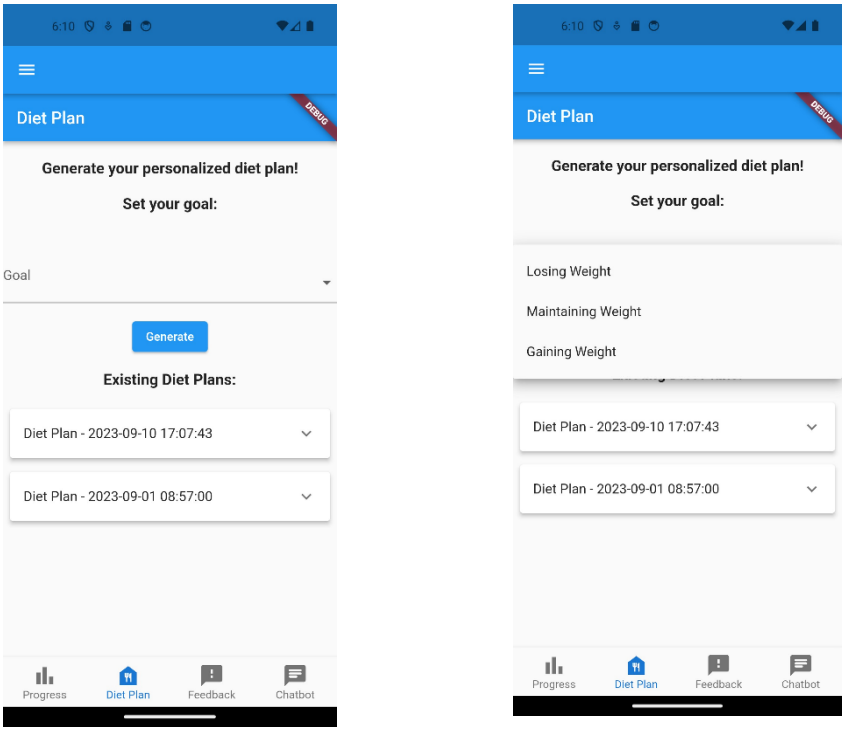



Figure 5.15 : Actual Progress Screen

6. Diet Plan Screen



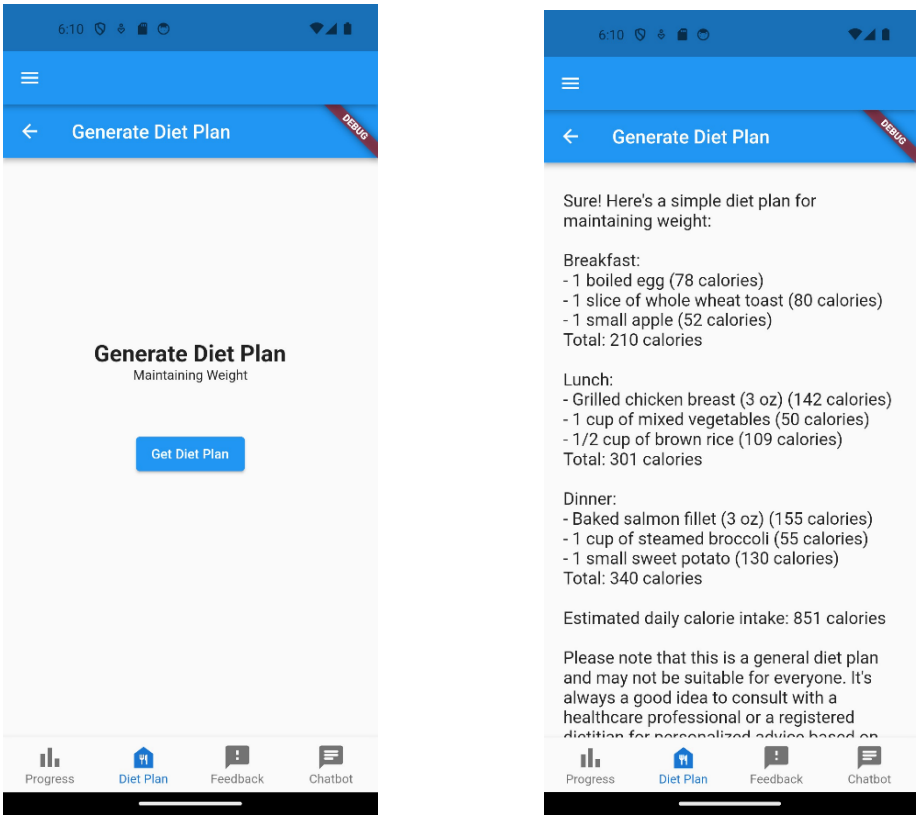




Figure 5.16: Actual Diet Plan Screen

7. Feedback Screen

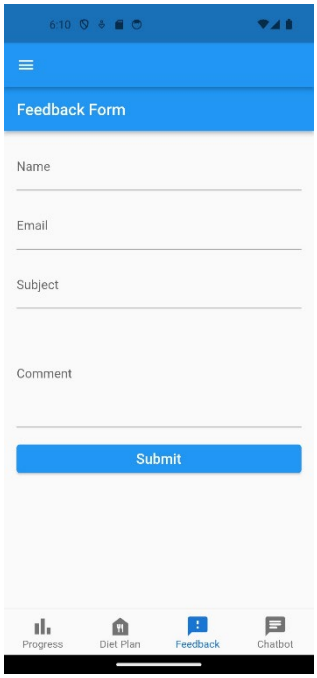



Figure 5.17 : Actual Feedback Screen

8. Chatbot Screen

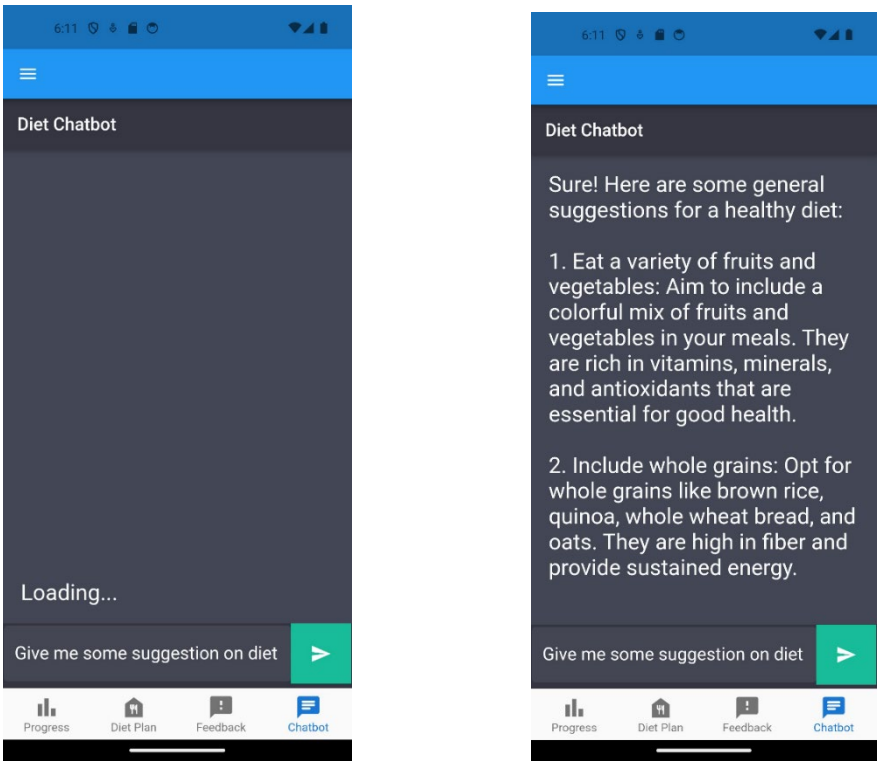



Figure 5.18 : Actual Chatbot Screen

# CHAPTER 6

# IMPLEMENTATION

## 6.1 Introduction

This chapter provides a comprehensive overview of the project's implementation process. It outlines the integration of essential components into the Flutter-based Android mobile application by using Dart as the programming language. This is because Flutter has a wide range of widgets and libraries that will ease the development on user interface.

Furthermore, the chapter presents a detailed implemented of API endpoint which is the integration of the OpenAI API into the system. These endpoints facilitate communication between the application and the OpenAI API, allowing for the retrieval of vital information related to personalized diet plans and nutrition.

## 6.2 Flutter Libraries

Flutter's Material Design significantly enhances the diet app's user interface, offering a versatile array of user-friendly components that elevate the overall user experience. This renowned UI library empowers developers to create visually appealing and coherent applications, seamlessly integrating them into users' daily routines. Within this project, Material Design's UI elements are harnessed to enhance both aesthetics and functionality.

Drawer and bottom navigation bar are implemented to ensure a seamless and intuitive user experience. The drawer provides easy access to essential app functions and navigation options, making it effortless for users to explore various features and sections. Additionally, the bottom navigation bar offers a visually cohesive and user-friendly way to switch between key app destinations, enhancing navigation efficiency.

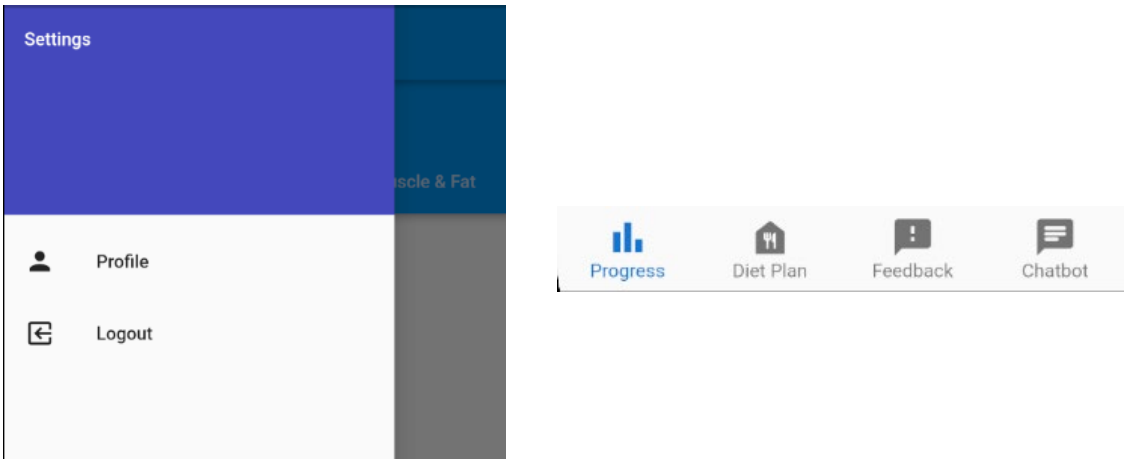



Figure 6.1 : Drawer and Bottom Navigation Bar

Text inputs feature prominently in various user interactions, facilitating activities like sign-up, sign-in, and profile updates. These inputs streamline data collection, making it effortless for users to provide essential information such as usernames and personal details. Dropdown features also used for setting user's personal diet plan goal.

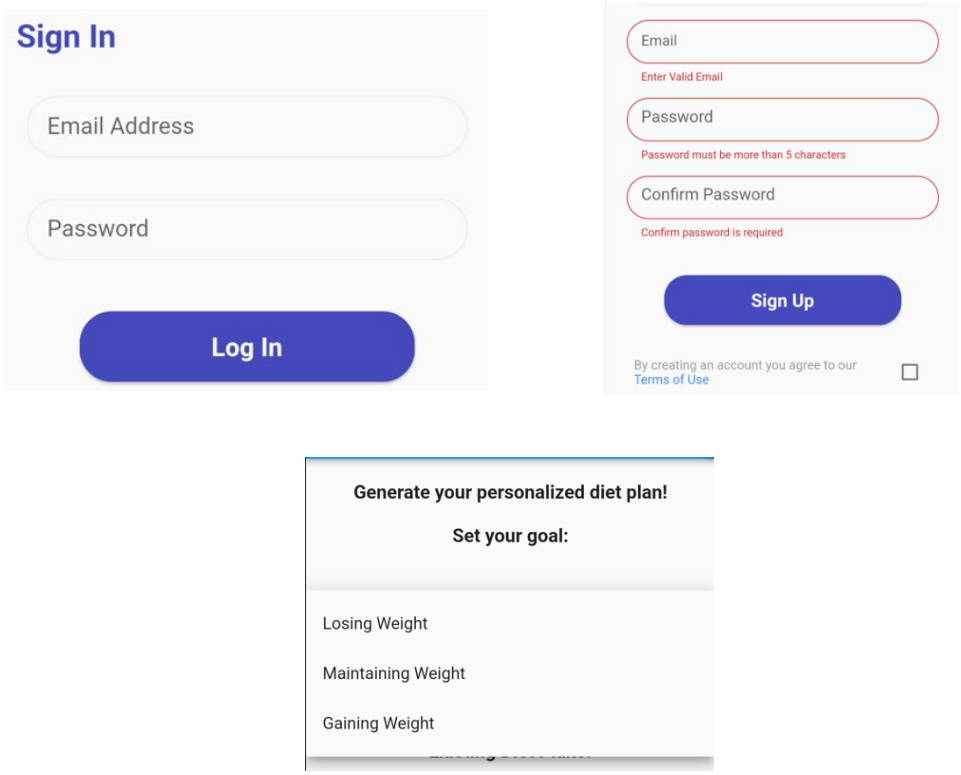



Figure 6.2 : Flutter Text Input Samples

Progress bars offer real-time feedback, visualizing ongoing tasks like during the loading of user authentication. Users can track progress, ensuring transparency and engagement.

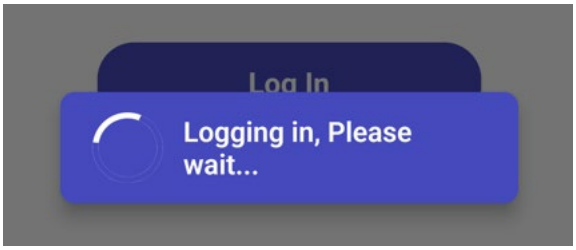



Figure 6.3 : Loading progress bar

Floating Action Buttons provide quick access to vital actions, enabling users to effortlessly add their current fat mass and lean mass for recording their health attributes. This convenience fosters active engagement.

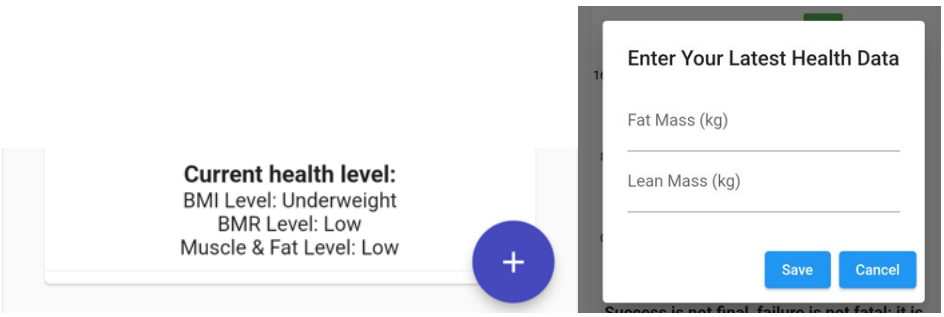



Figure 6.4 : Floating Action Button Sample

By leveraging Material Design's UI elements, the app aims to deliver a visually appealing, user-friendly, and indispensable tool for managing diets and promoting better health.

## 6.3 Firebase

### 6.3.1 Implementation of Firebase Authentication

Firebase Authentication serves as a robust tool for verifying users through various means. It not only facilitates user authentication but also offers features like email and phone number verification, as well as password reset functionality, which are readily accessible. Additionally, Firebase Authentication extends support for third-party authentication methods,

enhancing its versatility. One of its notable security features is the automatic generation of random user IDs during user registration and the secure hashing of passwords before storage in the cloud repository. Within this android mobile application, all Firebase Authentication functionalities have been encapsulated within the User Auth Service, ensuring convenient accessibility from different views. The mobile app exclusively employed email and password authentication methods for user verification. The Firebase Authentication was mainly implemented in two functions which are sign in and sign up.

### 6.3.2 Sign in

Users can log in using the authentication credentials that they have sign up previously. Once again, Firebase Authentication's login methods will handle the authentication process automatically by verifying the user's inputs and display appropriate error messages accordingly. If login success, users will then be redirected to the Progress screen and able to access to the other features of the application.

```
static Future<dynamic> loginWithEmailAndPassword(
    String email, String password) async {
  try {
    auth.UserCredential result = await auth.FirebaseAuth.instance
        .signInWithEmailAndPassword(email: email, password: password);
    DocumentSnapshot<Map<String, dynamic>> documentSnapshot = await firestore
        .collection(usersCollection)
        .doc(result.user?.uid ?? '')
        .get();
    User? user;
    if (documentSnapshot.exists) {
      user = User.fromJson(documentSnapshot.data() ?? {});
    }
    return user;
  } on auth.FirebaseAuthException catch (exception, s) {
    debugPrint('$exception$s');
    switch ((exception).code) {
      case 'invalid-email':
        return 'Email address is malformed.';
      case 'wrong-password':
        return 'Wrong password.';
      case 'user-not-found':
        return 'No user corresponding to the given email address.';
      case 'user-disabled':
```

Figure 6.5 : Sign in with firebase authentication code segment-1

```
on<LoginWithEmailAndPasswordEvent>((event, emit) async {
  dynamic result = await FireStoreUtils.loginWithEmailAndPassword(
      event.email, event.password);
  try {
    result = await FireStoreUtils.loginWithEmailAndPassword(
      event.email, event.password);
    print('Login Result: $result');
  } catch (e) {
    print('Unknown error occurred during authentication: $e');
    result = null; // Set the result to null in case of an unknown error
  }
  if (result != null && result is User) {
    user = result;
    emit(AuthenticationState.authenticated(user!));
  } else if (result != null && result is String) {
    emit(AuthenticationState.unauthenticated(message: result));
  } else {
    emit(const AuthenticationState.unauthenticated(
        message: 'Login failed!!!, Please try again.'));
  }
});
```

Figure 6.6 : Sign in with firebase authentication code segment-2

### 6.3.3 Sign Up

If the user does not have an account, user is required to register for a new account. Users required to click on “Sign Up” button in the onboarding screen. Then, users will be redirected to an interface which prompts to enter the email address, password, age, weight, height, gender, medical history and food preferences. Once all the required information is filled correctly user clicks on the “Sign Up” button where the Firebase Authentication’s sign-up function will be called to handle to user sign up process. Alert will be display to user indicating the account created successfully whereby if the account credential is invalid then error message will be prompted.

```
static Future<String?> createNewUser(User user) async => await firestore
    .collection(usersCollection)
    .doc(user.userID)
    .set(user.toJson())
    .then((value) => null, onError: (e) => e);

static signUpWithEmailAndPassword(
    {required String emailAddress,
    required String password,
    Uint8List? imageData,
    firstName = 'Anonymous',
    lastName = 'User',
    String? age,
    String? weight,
    String? height,
    String? gender,
    String? medicalHistory,
    String? foodPreferences}) async {
  try {
    auth.UserCredential result = await auth.FirebaseAuth.instance
        .createUserWithEmailAndPassword(
            email: emailAddress, password: password);
```

Figure 6.7 : Sign up with firebase authentication code segment-1

```
on<SignupWithEmailAndPasswordEvent>((event, emit) async {
  dynamic result = await FireStoreUtils.signUpWithEmailAndPassword(
      emailAddress: event.emailAddress,
      password: event.password,
      imageData: event.imageData,
      firstName: event.firstName,
      lastName: event.lastName,
      age: event.age,
      weight: event.weight,
      height: event.height,
      gender: event.gender,
      medicalHistory: event.medicalHistory,
      foodPreferences: event.foodPreferences);
  if (result != null && result is User) {
    user = result;
    emit(AuthenticationState.authenticated(user!));
  } else if (result != null && result is String) {
    emit(AuthenticationState.unauthenticated(message: result));
  } else {
    emit(const AuthenticationState.unauthenticated(
        message: 'Couldn\'t sign up'));
  }
```

Figure 6.8 : Sign up with firebase authentication code segment-2

### 6.3.4 Implementation of Firebase Firestore Database

The incorporation of Firebase Firestore Database into the app's architecture has played a crucial role in providing users with a dynamic and responsive experience. This cloud-based NoSQL database seamlessly stores and retrieves data, ensuring that user's profile information, diet plan, progress tracking records and feedbacks are securely managed and readily accessible.

Firebase Firestore's real-time data synchronization is fundamental to the app's functionality. As users interact with the app, any updates or changes they make are instantaneously reflected, creating a dynamic and responsive environment. For example, when users record their health data, this data is immediately stored in the cloud, ensuring that they always have access to the most up-to-date information. Furthermore, Firebase Firestore's robust security features ensure that sensitive information is protected from unauthorized access, instilling trust and confidence in our users.

In conclusion, the integration of Firebase Firestore Database has significantly enhanced the functionality and reliability of the app. It empowers the app to offer real-time updates, secure data management, and a seamless user experience, ultimately contributing to the success and user satisfaction of this diet application.

#### 6.3.4.1 Profile

When the authenticated user is navigated to the profile screen, the screen will retrieve the current user's profile information on the interface. If the user wanted to modify the profile information, the user presses on the edit icon button and fill in the updated latest information of the user. Once the user finished updating the information, user presses on the save button. This will trigger the **_saveChanges()** function in order to store the latest user profile information into Firestore Database.

```
Future<void> _saveChanges() async {
  try {
    // Query for the document based on the email field
    QuerySnapshot querySnapshot = await FirebaseFirestore.instance
        .collection('users')
        .where('email', isEqualTo: _editedUser.email)
        .get();

    if (querySnapshot.docs.isNotEmpty) {
      // Get the first document found (you might want to add more checks here)
      DocumentSnapshot documentSnapshot = querySnapshot.docs.first;

      // Update the document with the new data
      await documentSnapshot.reference.update({
        'gender': _editedUser.gender,
        'weight': _editedUser.weight,
        'height': _editedUser.height,
        'medicalHistory': _editedUser.medicalHistory,
        'foodPreferences': _editedUser.foodPreferences,
      });
```

Figure 6.9 : Profile with firestore database code segment

### 6.3.4.2 Progress

Once user is authenticated, progress screen will be the first to be navigated. There are a total of 3 progress tabs which are BMI, BMR and Fat & Muscle percentage record. The calculation formula is implemented in the progress tracker function as below in order to keep track of user's health status.

```
double calculateBMI(double heightInCM, double weightInKG) {
  // BMI Formula: weight (kg) / (height (m))^2
  double heightInM = heightInCM / 100.0;
  double bmi = weightInKG / (heightInM * heightInM);
  return bmi;
}

double calculateBMR(double weightInKG, double heightInCM, int age, String gender) {
  if (gender == 'Male') {
    // BMR Formula for males: 10 * weight (kg) + 6.25 * height (cm) - 5 * age + 5
    return 10 * weightInKG + 6.25 * heightInCM - 5 * age + 5;
  } else {
    // BMR Formula for females: 10 * weight (kg) + 6.25 * height (cm) - 5 * age - 161
    return 10 * weightInKG + 6.25 * heightInCM - 5 * age - 161;
  }
}

double calculateFatPercentage(double weightInKG, double fatMass) {
  return (fatMass / weightInKG) * 100;
}

double calculateMuscleMass(double weightInKG, double fatMass, double leanMass) {
  return weightInKG - fatMass - leanMass;
}
```

Figure 6.10: Progress with firestore database code segment-1

The application will call the function **getUserProgressStream()** once user is navigated to progress screen to retrieve the current user's progress data by date in descending order from the Firestore Database.

```
Stream<List<UserProgress>> getUserProgressStream(String dataType) {
  return FirebaseFirestore.instance
      .collection('userProgress')
      .where('email', isEqualTo: widget.user.email)
      .orderBy('date')
      .snapshots()
      .map((snapshot) => snapshot.docs.map((doc) => UserProgress(
            date: doc['date'].toDate(),
            bmi: doc['bmi'],
            bmr: doc['bmr'],
            muscleMass: doc['muscleMass'],
            fatPercentage: doc['fatPercentage'],
            email: doc['email'],
          )).toList());
}
```

Figure 6.11 : Progress with firestore database code segment-2

If the user wants to record a new progress data, the user presses on the floating 'Add' button and fills in the health information accordingly. Then the system will calculate the user's BMI, BMR, muscle mass and fat percentage by using the calculation function as stated above. Next, the system will store the calculated values in separated variables and the variables will be added to **userProgress** collection in Firestore Database as the **recordUserProgress()** function is called.

```
Future<void> recordUserProgress(double muscleMass, double fatPercentage) async {
  double bmi = calculateBMI(widget.user.height, widget.user.weight);
  double bmr = calculateBMR(widget.user.weight, widget.user.height, widget.user.age, widget.user.gender);

  DateTime currentDate = DateTime.now();

  UserProgress progress = UserProgress(
    date: currentDate,
    bmi: bmi,
    bmr: bmr,
    muscleMass: muscleMass,
    fatPercentage: fatPercentage,
    email: widget.user.email,
  );

  await FirebaseFirestore.instance.collection('userProgress').add({
    'date': progress.date,
    'bmi': progress.bmi,
    'bmr': progress.bmr,
    'muscleMass': progress.muscleMass,
    'fatPercentage': progress.fatPercentage,
    'email': progress.email,
  });
```

Figure 6.12: Progress with firestore database code segment-3

#### 6.3.4.3 Generate Diet Plan

When the user navigates to the view diet plan screen, the function loadDietPlans() will be invoked. The existing diet plans of the current user will be retrieved from the dietPlan collection in Firestore Database. The interface will then prompt the retrieved existing diet plan to the user.

```
// Function to load the user's diet plans from Firestore
void loadDietPlans() {
  FirebaseFirestore.instance
      .collection('dietPlan')
      .where('userEmail', isEqualTo: widget.user.email)
      .orderBy('timestamp', descending: true)
      .get()
      .then((QuerySnapshot querySnapshot) {
    setState(() {
      dietPlans = querySnapshot.docs;
    });
  }).catchError((error) {
    print('Error loading diet plans: $error');
  });
}
```

Figure 6.13: Generate diet plan with firestore database code segment-1

If the user wants to delete the existing current diet plan, the user may press on the specific diet plan to view and look up for delete icon button. Once user presses on the delete icon button, the delete diet plan function will be

```
// Delete the diet plan from Firestore
FirebaseFirestore.instance
    .collection('dietPlan')
    .doc(dietPlan.id)
    .delete()
    .then((value) {
  // Reload the diet plans
  loadDietPlans();
  ScaffoldMessenger.of(context).showSnackBar(
    const SnackBar(
      content: Text('Diet plan deleted.'),
    ), // SnackBar
  );
}).catchError((error) {
  print('Error deleting diet plan: $error');
  ScaffoldMessenger.of(context).showSnackBar(
    const SnackBar(
      content: Text('Error deleting diet plan.'),
```

Figure 6.14: Generate diet plan with firestore database code segment-2

invoked. As a result, the chosen diet plan will be deleted from the system and in Firestore Database as well.

If the user wants to generate a new diet plan, the user has to choose their goal preferences then press on the 'Generate' button. The user will then be navigated to the generate diet plan screen. Next, the user will press the 'Get Diet Plan' button in order for the system to generate a latest diet plan for the user based on the user latest progress data which is retrieved by the function **getUserLatestProgressStream()**. Once the diet plan is generated, the function **storeResponseInFirestore()** will be invoked and the latest diet plan generated will be stored in **dietPlan** collection in Firestore Database.

```
Stream<UserProgress?> getUserLatestProgressStream(String email) {
  return FirebaseFirestore.instance
      .collection('userProgress')
      .where('email', isEqualTo: email)
      .orderBy('date', descending: true)
      .limit(1)
      .snapshots()
      .map((snapshot) {
        if (snapshot.docs.isNotEmpty) {
          final doc = snapshot.docs[0];
          return UserProgress(
            date: doc['date'].toDate(),
            bmi: doc['bmi'],
            bmr: doc['bmr'],
            muscleMass: doc['muscleMass'],
            fatPercentage: doc['fatPercentage'],
            email: doc['email'],
          );
        } else {
          return null; // Return null if no progress found
        }
      });
}
```

```
Future<void> storeResponseInFirestore(String userEmail, String response) async {
  try {
    await FirebaseFirestore.instance.collection('dietPlan').add({
      'userEmail': userEmail,
      'response': response,
      'timestamp': FieldValue.serverTimestamp(),
    });
    debugPrint('Response stored in Firestore');
  } catch (e) {
    debugPrint('Error storing response in Firestore: $e');
  }
}
```

Figure 6.15: Store diet plan in firestore database code segment

#### 6.3.4.4 Feedback

When user is navigated to the feedback screen, the user is required to fill in the feedback form. Once the user filled the feedback form, the user will press on submit button. This will invoke the **_submitFeedback()** function and will add the feedback information to the feedback collection in the Firestore Database.

```
Future<void> _submitFeedback() async {
  if (_formKey.currentState!.validate()) {
    setState(() {
      _isSubmitting = true;
    });

    try {
      final user = FirebaseAuth.instance.currentUser;
      if (user != null) {
        final feedbackData = {
          'name': _nameController.text,
          'email': _emailController.text,
          'subject': _subjectController.text,
          'comment': _commentController.text,
        };

        final emailExists = await _checkEmailExists(_emailController.text);
        if (emailExists) {
          _showErrorSnackBar('You have already submitted a form with this email.');
        } else {
          await FirebaseFirestore.instance.collection('feedback').add(feedbackData);
          _showSuccessSnackBar('Feedback submitted successfully!');
          _resetForm(); // Clear the form after successful submission
        }
```

Figure 6.16: Feedback in firestore database code segment

## 6.4 Implementation of OpenAI API

The implementation of the OpenAI API plays a pivotal role in our project. This integration empowers the diet app to provide intelligent and context-aware responses to user inquiries and prompts. Through the utilization of the OpenAI API, the app gains access to state-of-the-art language models, particularly the gpt-3.5-turbo model. This model serves as the AI diet consultant, offering valuable insights and information related to diet. The integration involves constructing and sending API requests, including user-generated prompts and system messages, to the OpenAI service. The responses received from the API are then processed and presented to the user, enhancing the app's capabilities in addressing user queries and assisting with dietary concerns. To facilitate the integration of the Flutter mobile application with the OpenAI API, a subscription fee must be remitted, and the generated API key should be stored in a .env file, which is then declared in the pubspec.yaml's flutter assets.

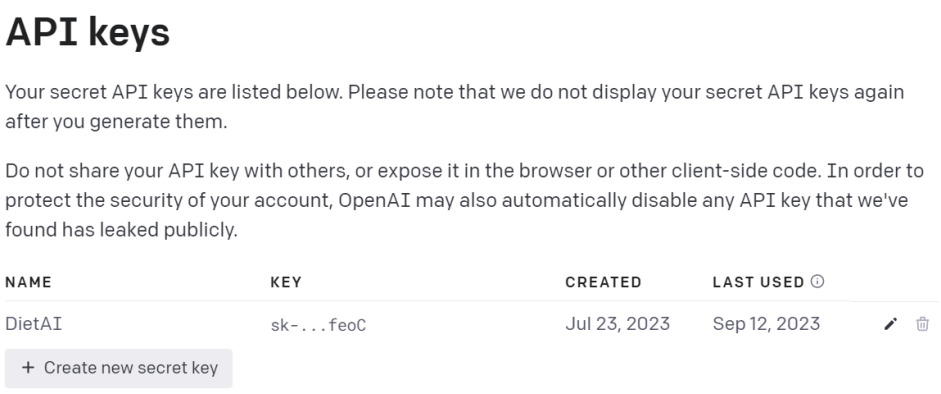



Figure 6.17: Example of OpenAI API key

### 6.4.1 Generate diet plan

The diet plan generation function works by setting up the prompt content at the backend code so that the system does not need to prompt any extra or additional input from user to make sure the diet plan generation is consistent. The diet plan generation does take user's selected goal preferences, physical attributes, food preferences, medical conditions and latest health progress data into consideration as the information is prompted to the OpenAI API which is stored in the **promptContent**.

```
Future<void> completionFun() async { // Mark the method as async
  setState(() => responseTxt = 'Loading...');
  final userProgress = await getUserLatestProgressStream(widget.user.email).first;
  final promptContent = 'Please help create a $selectedGoal diet plan that consists of breakfast, lunch, and dinner only.
  promptController.text = promptContent;
```

Figure 6.18: Generate diet plan's OpenAI implementation -1

Besides, system message is also set as 'Generate a diet plan for goal: $selectedGoal', so that the response behaviour will be based more specifically on the goal related information.

```
final response = await http.post(
  Uri.parse('https://api.openai.com/v1/chat/completions'),
  headers: {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer ${dotenv.env['token']}',
  },
  body: jsonEncode(
    {
      "model": "gpt-3.5-turbo",
      "messages": [
        {
          "role": "system",
          "content": "Generate a diet plan for goal: $selectedGoal",
        },
        {
          "role": "user",
          "content": promptController.text,
        },
      ],
      "max_tokens": 500,
      "temperature": 0,
      "top_p": 1,
    },
  ),
```

Figure 6.19: Generate diet plan's OpenAI implementation -2

### 6.4.2 Chatbot

The chatbot feature works by setting up the OpenAI API by setting up the messages content into “You are helpful diet consultant” so that the response given will be based on a diet consultant perspective to prevent off topic in response. For this feature, the application will prompt for user’s input for a question related to diet topic. The question asked can be in a wide range and does not have any limitations on it but the response will sticks around with diet or health related topic.

```
final response = await http.post(
  Uri.parse('https://api.openai.com/v1/chat/completions'),
  headers: {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer ${dotenv.env['token']}',
  },
  body: jsonEncode(
    {
      "model": "gpt-3.5-turbo", // Use the gpt-3.5-turbo model
      "messages": [
        {
          "role": "system",
          "content": "You are a helpful diet consultant.", // Add a system message to set the behavio
        },
        {
          "role": "user",
          "content": promptController.text, // Use the user's input as the prompt
        },
      ],
      "max_tokens": 500,
      "temperature": 0,
      "top_p": 1,
    },
```

Figure 6.20: Chatbot’s OpenAI implementation

## 6.5 Summary

This chapter discussed the implementation of the actual prototype using Flutter and Dart. It discusses the integration of Flutter's Material Design for improved user interface, Firebase for user authentication, and Firestore Database for secure data management. Additionally, the chapter covers the implementation of the OpenAI API, enabling the app to provide personalized diet plans and context-aware responses through an AI chatbot. These technical aspects enhance the app's functionality, delivering a user-friendly and dynamic diet management solution.

# CHAPTER 7

# TESTING AND EVALUATION

## 7.1 Introduction

This chapter provides an overview of the system's test processes and review of the testing outcomes. Unit and user acceptance testing were incorporated in the testing processes for this project. To assess the alignment between the test cases, use cases, and functional requirements detailed in Chapter 4, a traceability matrix was generated. This matrix serves as a tool for tracking how well the test cases correspond to the specified use cases and functional requirements.

## 7.2 Traceability between Use Cases, Functional Requirements and Test Cases

For software developers, conducting comprehensive testing is an integral aspect of ensuring the system's functionality and reliability. Testing methodologies encompass various techniques and serve distinct purposes, such as unit testing, which assesses the performance of specific components within the system. However, as the number of test cases grows, developers may encounter challenges in managing and potentially generating redundant or superfluous test cases, which can be detrimental to the system's quality. To assist readers in gaining a broad comprehension of the objectives of individual test cases, this chapter incorporates straightforward traceability matrices. These matrices establish clear connections between test cases, relevant functional specifications, and use cases, enabling a precise understanding of each test case's purpose and alignment with the project's goals.

### 7.2.1 Use Case Table

Table 7.1 Use Cases

| Use Case ID | Use Case Name |
|---|---|
| UC001 | Login Account |
| UC002 | Register Account |
| UC003 | Modify Profile |
| UC004 | Record Progress |
| UC005 | View Progress |
| UC006 | Generate Diet Plan |
| UC007 | View Diet Plan |
| UC008 | Use Chatbot |
| UC009 | Receive feedback |

### 7.2.2 Functional Requirement Table

Table 7.2: Functional Requirements

| Functional Requirement ID | Functional Requirement Statement | Related Use Case ID | Related Test Case ID |
|---|---|---|---|
| FR001 | The application shall be able to allow user to register and create a user profile by providing basic information. | UC002 | UTC006-UTC012 |
| FR002 | The application shall allow user to modify their user profiles. | UC003 | UTC014-UTC016 |
| FR003 | The application shall be able to generate a personalized diet plan based on user's goals and food preferences. | UC006, UC007 | UTC023-UTC024 |
| FR004 | The application shall provide basic information about the foods included in the diet plan such as calorie value. | UC006, UC007 | UTC021-UTC022 |

| | | | |
|---|---|---|---|
| FR005 | The application shall allow users to record their progress toward their goals by tracking their health attributes. | UC004 | UTC019-UTC020 |
| FR006 | The application shall provide visual representations of progress, such as graphs. | UC005 | UTC017-UTC018 |
| FR007 | The application shall be able to allow users to log their lean mass, fat mass, height and weight manually in daily basis. | UC004 | UTC019-UTC020 |
| FR008 | The application shall have a chatbot feature that allows users to ask questions about diet related problems and receive personalized responses. | UC008 | UTC027-UTC028 |
| FR009 | The application shall be able to receive feedback from user to further improvise on the application. | UC009 | UTC025-UTC026 |

### 7.3 Unit Testing

A technique used to ensure that each individual software component achieves the features and functionalities specified by its pre-defined set of criteria is known as unit testing. The complete software system was divided into separate units, with appropriate test cases linked to each of the aforementioned components. As a result, software system faults and defects can be found similarly to a bottom-to-top strategy. As a result, it takes less time to debug problems in the code base, which in turn speeds up development because the code base can be reduced to the source code of a single unit.

Table 7.3: Unit testing test cases

| Test Case ID | Test Case Name | Test Case Description | Related Use Case | Status |
|---|---|---|---|---|
| UTC001 | Test login with correct | Examine whether the system will show login | Login Account | Pass |

| | | | | |
|---|---|---|---|---|
| | credential | success | | |
| UTC002 | Test login with correct email but invalid password | Examine whether the system will show error message | Login Account | Pass |
| UTC003 | Test login with invalid email | Examine whether the system will show error message | Login Account | Pass |
| UTC004 | Test login with an empty email | Examine whether the system will show error message | Login Account | Pass |
| UTC005 | Test login with an empty password | Examine whether the system will show error message | Login Account | Pass |
| UTC006 | Test sign up with valid information | Examine whether the system will show sign up success | Register Account | Pass |
| UTC007 | Test sign up with all empty fields | Examine whether the system will show error message | Register Account | Pass |
| UTC008 | Test sign up with valid information but with empty first name and last name | Examine whether the system will show error message | Register Account | Pass |
| UTC009 | Test sign up with valid information but with empty email. | Examine whether the system will show error message | Register Account | Pass |
| UTC010 | Test sign up with valid information but with invalid email. | Examine whether the system will show error message | Register Account | Pass |

| | | | | |
|---|---|---|---|---|
| UTC011 | Test sign up with valid information but with empty password | Examine whether the system will show error message | Register Account | Pass |
| UTC012 | Test sign up with valid information but with empty confirm password | Examine whether the system will show error message | Register Account | Pass |
| UTC013 | Test user signs out current user | Examine whether the system will end the current user's session | - | Pass |
| UTC014 | Test get current user's profile information | Examine whether the system able to retrieve user's profile information | Modify Profile | Pass |
| UTC015 | Test modify user's profile | Examine whether the system able to update user's profile information | Modify Profile | Pass |
| UTC016 | Test modify user's profile using invalid input | Examine whether the system able to show error message | Modify Profile | Fail |
| UTC017 | Test get user progress information | Examine whether the system able to retrieve user's progress information | View Progress | Pass |
| UTC018 | Test get user progress information if no progress information found. | Examine whether the system able to show error message | View Progress | Pass |

| | | | | |
|---|---|---|---|---|
| UTC019 | Test add user progress | Examine whether the system able to retrieve user's latest progress information | Record Progress | Pass |
| UTC020 | Test add user progress with invalid value | Examine whether the system able to show error message. | Record Progress | Pass |
| UTC021 | Test view specific existing diet plan | Examine whether the system able to retrieve user's specific existing diet plan information. | View diet plan | Pass |
| UTC022 | Test delete specific existing diet plan | Examine whether the system able to delete the chosen existing diet plan. | View diet plan | Pass |
| UTC023 | Test generate diet plan | Examine whether the system able to prompt the diet plan information and update the latest diet plan to the database | Generate diet plan | Pass |
| UTC024 | Test generate diet plan without an existing progress data | Examine whether the system able to show message. | Generate diet plan | Pass |
| UTC025 | Test record feedback | Examine whether the system able to retrieve user's feedback | Receive feedback | Pass |
| UTC026 | Test record feedback with blank input | Examine whether the system able to show message. | Receive feedback | Pass |

| | | | | |
|---|---|---|---|---|
| UTC027 | Test diet chatbot with a valid question | Examine whether the system able to prompt response according to user input | Use chatbot | Pass |
| UTC028 | Test diet chatbot with a blank input | Examine whether the system able to show error message. | Use chatbot | Pass |

### 7.3.1 Unit Test Results

The unit test results can be found in Appendix B.

## 7.4 Integration Testing

Integration tests look at interoperability between components to make sure modules can interact and communicate with one another correctly. Manual integration testing is done to ensure sure all of the shown data is appropriate and accurate.

### 7.4.1 Test Cases Table of Integration Testing

Table 7.4: Integration Testing test cases

| **Test Case ID** | **Test Case Name** | **Test Case Description** | **Status** |
|---|---|---|---|
| ITC001 | Test login screen | Examine whether the user can navigate to login page | Pass |
| ITC002 | Test progress screen if user is authenticated. | Examine whether the user can navigate to progress page if user is authenticated. | Pass |
| ITC003 | Test progress screen if user is not authenticated. | Examine whether the user is redirect back to login page if the user is not authenticated. | Pass |
| ITC004 | Test log out | Examine whether the user is removed from session and | Pass |

| | | | |
|---|---|---|---|
| | | redirect to the onboarding screen. | |
| ITC005 | Test sign up screen | Examine whether user can navigate to sign up screen. | Pass |
| ITC006 | Test submit sign up | Examine whether user can navigate to user's progress screen | Pass |
| ITC007 | Test user profile screen | Examine whether the user can navigate to user profile screen after the user is authenticated | Pass |
| ITC008 | Test update profile information | Examine whether the new information is updated to the backend | Pass |
| ITC009 | Test view diet plan screen | Examine whether the user can navigate to view diet plan screen after the user is authenticated. | Pass |
| ITC010 | Test generate diet plan screen | Examine whether the user can navigate to generate diet plan screen after pressing on "Generate" button | Pass |
| ITC011 | Test add diet plan | Examine whether the diet plan information is added to database | Pass |
| ITC012 | Test delete diet plan | Examine whether the diet plan information is deleted from database | Pass |
| ITC013 | Test add progress | Examine whether the add progress information is added to the backend | Pass |
| ITC014 | Test feedback screen | Examine whether user can navigate to feedback screen after user is authenticated | Pass |

| | | | |
|---|---|---|---|
| ITC015 | Test add feedback | Examine whether the feedback information is added to the backend | Pass |
| ITC016 | Test chatbot screen | Examine whether user can navigate to chatbot after the user is authenticated | Pass |

### 7.4.2 Integration Testing Result

The integration test results can be found in Appendix C.

## 7.5 User Acceptance Test

User acceptance testing (UAT) distinguishes itself from previous tests, such as integration testing and unit testing. Unlike these tests that primarily focus on bug discovery and system functionality, UAT centers on evaluating user satisfaction and the system's effectiveness in addressing specific issues. The absence of bugs in a system doesn't automatically guarantee success in UAT because what users prioritize is the system's capability to solve their problems. The primary objective of developing this application is to provide users with a comprehensive tool for managing their dietary habits and achieving their fitness goals. Additionally, this app aims to empower users to make healthier food choices, monitor their progress, and ultimately improve their overall well-being.

### 7.5.1 User Acceptance Test Plan

The user acceptance test was planned to be conducted with a wide range of users, as this application aims to cater to a diverse user base. Our goal was to ensure that users from all backgrounds and with varying needs could effectively utilize and benefit from the application. To achieve this, we designed multiple surveys to gather feedback from different user groups, accommodating the unique expectations and requirements of each category of users among UTAR Sungai Long Campus. As part of this project, five designated target users were invited to participate in the User Acceptance Testing (UAT) phase, following the provided test form located in Appendix D. Throughout the UAT phase, the

developer closely supervised the testing procedures and offered guidance to participants should they encounter any inquiries or concerns.

### 7.5.2 User Acceptance Test Cases

Table 7.5 User Acceptance Test Cases

| Test Case ID | Test Case Name | Test Case Description | Status |
|---|---|---|---|
| UAT001 | Test login with correct credential | Examine whether the system will show login success | Pass |
| UAT002 | Test login with invalid credential | Examine whether the system will show error message | Pass |
| UAT003 | Test sign up with valid information | Examine whether the system will show sign up success | Pass |
| UAT004 | Test sign up with invalid input | Examine whether the system will show error message | Pass |
| UAT005 | Test user signs out current user | Examine whether the system will end the current user's session | Pass |
| UAT006 | Test get current user's profile information | Examine whether the system able to retrieve user's profile information | Pass |
| UAT007 | Test modify user's profile | Examine whether the system able to update user's profile information | Pass |
| UAT008 | Test modify user's profile using invalid input | Examine whether the system able to show error message | Pass |
| UAT009 | Test get user progress information | Examine whether the system able to retrieve user's progress information | Pass |
| UAT010 | Test get user progress information if no progress information found. | Examine whether the system able to show error message | Pass |

| | | | |
|---|---|---|---|
| UAT011 | Test add user progress | Examine whether the system able to retrieve user's latest progress information | Pass |
| UAT012 | Test add user progress with invalid value | Examine whether the system able to show error message. | Pass |
| UAT013 | Test view specific existing diet plan | Examine whether the system able to retrieve user's specific existing diet plan information. | Pass |
| UAT014 | Test delete specific existing diet plan | Examine whether the system able to delete the chosen existing diet plan. | Pass |
| UAT015 | Test generate diet plan | Examine whether the system able to prompt the diet plan information and update the latest diet plan to the database | Pass |
| UAT016 | Test generate diet plan without an existing progress data | Examine whether the system able to show message. | Pass |
| UAT017 | Test record feedback | Examine whether the system able to retrieve user's feedback | Pass |
| UAT018 | Test record feedback with blank input | Examine whether the system able to show message. | Pass |
| UAT019 | Test diet chatbot with a valid question | Examine whether the system able to prompt response according to user input | Pass |
| UAT020 | Test diet chatbot with a blank input | Examine whether the system able to show error message. | Pass |

### 7.5.3 User Acceptance Test Result

The user acceptance test results can be found in Appendix D.

### 7.5.4 User Acceptance Test Result Summary

The user acceptance test for the Android mobile application took place among students at the UTAR Sungai Long Campus. 15 participants were involved in this assessment, and there was a constraint on the time available to engage with

the testers. Following the test, 8 questions were used in a feedback form to gather input from the testers. The number of participants and age group, and the average rating of each question are listed in the table below. The feedback form results can be found in Appendix E.

Table 7.6: UAT participants age group

| Age Group | Number of participants |
|---|---|
| 18-24 | 4 |
| 25-34 | 9 |
| 35-44 | 2 |

Table 7.7: UAT feedback result summary

| No | Question | Average Rating (1-5) |
|---|---|---|
| Q1 | What features of this application do you find most valuable? (Select all that apply) | 5 (AI Diet Chatbot) |
| Q2 | I think that I would like to use this application to keep a good diet. | 4.9 |
| Q3 | I found the application unnecessarily complicated. | 3.6 |
| Q4 | The application was easy to use. | 4.7 |
| Q5 | I think that there are too many inconsistencies in this application. | 2.2 |
| Q6 | How would you rate your overall satisfaction with this application? | 4.7 |
| Q7 | How likely are you to recommend this application to a friend or colleague? | 4.67 |
| Q8 | I am looking forward for using this application further | 4.7 |

## 7.6 Summary

This chapter reveals the outcomes of various tests. Positive results from unit tests, integration tests, and user acceptance tests indicate that the application is

fully functional and ready for use. Notably, the user acceptance test yielded positive average results, affirming the application's usability and effectiveness.

# CHAPTER 8

# CONCLUSION AND RECOMMENDATION

## 8.1 Introduction

As we reach the culmination of this comprehensive journey through the development of the Android mobile application, it is essential to reflect on the milestones achieved and the insights gained along the way. This concluding chapter encapsulates the essence of the project, which is dedicated to addressing critical issues in modern lifestyles, such as imbalanced diets, increasing indiscipline, and mental health concerns.

Throughout this endeavor, the objectives were outlined, and a comprehensive solution was proposed to leverage technology to promote healthier living. The implementation process followed a meticulous methodology, combining research and development approaches. A toolkit of essential tools and software was harnessed, from Axure RP for prototyping to Firebase for robust data management. The project's journey was meticulously planned, as reflected in the Work Breakdown Structure (WBS) and the Gantt Chart, which guided through each phase, from planning to design, development, and testing.

The initial project specification delved into collecting valuable data about the target audience's demographics, dietary habits, and their expectations from the application. This guided the formulation of both functional and non-functional requirements, culminating in detailed system use cases and interface flow diagrams.

The design phase brought the application to life with well-structured architecture, data models, and user interfaces. A robust system database was envisioned, enriched by a detailed data dictionary, and user interactions were visualized through activity diagrams. Next, the implementation stage saw the integration of Flutter libraries, Firebase authentication, and Firestore database

management. These elements form the backbone of the application's functionality, ensuring secure sign-in processes, seamless data storage, and retrieval.

The crowning achievement of the project was the integration of the OpenAI API, enabling the application to offer intelligent dietary guidance and responsive chatbot interactions. Testing and evaluation provided the assurance of quality, with positive results from unit testing, integration testing, and user acceptance testing, affirming the application's reliability and functionality.

In summary, this journey through the project has been a testament to the power of technology in addressing contemporary health challenges. The application aspires to make a positive impact on the lives of its users, promoting balanced diets, discipline, and mental well-being. As we conclude, we look forward to the potential growth and impact of the diet consultant application in promoting healthier lifestyles and fostering a brighter, more health-conscious future. This chapter discussed about the objective achieved, limitations of project and recommendations for future improvement.

## 8.2 Objective Achievement

The following describes how the project's objectives from Chapter 1 were fulfilled:

The first project objective is to investigate diet and health related data. The project embarked on an extensive investigation of diet and health-related data, underpinning the development of the proposed mobile. This thorough exploration provided a wealth of valuable insights and knowledge, forming the bedrock upon which the project's subsequent phases were built. It ensured that the application's functionalities were grounded in a robust understanding of the complexities of diet and health.

The second project objective was to conduct a business analysis based on existing application that is similar with this project. Initial phase of the

project involved conducting a comprehensive business analysis by comparing similar applications in the domain. This process yielded invaluable findings, enabling the project to gain profound insights into market dynamics, user expectations, and potential avenues for innovation. The thorough business analysis equipped the project with a strategic advantage, guiding its decisions and ensuring alignment with user needs.

The third project objective is to develop an android based mobile application that assist an individual to improve their diet. The core objective of developing an Android-based mobile application aimed at assisting individuals in enhancing their dietary habits has been successfully realized. The resultant application stands as a testament to the project's dedication to promoting healthier lifestyles and dietary choices. It encapsulates a wealth of knowledge and research, providing users with a user-friendly platform for dietary improvement.

The last project objective is to conduct an evaluation of the built mobile application with the target user. The project culminated in the user-centric evaluation of the developed mobile application. Through comprehensive user acceptance testing, the application's reception among the target user demographic was assessed. The results of this evaluation were overwhelmingly positive.

## 8.3 Limitations

While the primary objectives of the project were achieved, several limitations were identified during its course. The first limitation of the Android-based diet consultant application lies in the challenge of sourcing raw data for lean mass and fat mass measurements. Accurate and up-to-date information on these vital health indicators can be elusive, as they often require specialized equipment or medical assessments. This limitation underscores the reliance on user-generated data or third-party sources, potentially introducing variability and imprecision in the app's recommendations. Addressing this limitation would necessitate

exploring innovative data collection methods or partnerships with fitness and health monitoring devices.

The second limitation was the chatbot versatility beyond diet and health topics. While the diet consultant application boasts a chatbot feature for user interactions, it is constrained by its limited scope of responding to inquiries exclusively related to diet and health. This limitation restricts the chatbot's versatility, as users may seek information or engage in conversations on a wide range of topics. Expanding the chatbot's capabilities to address broader queries and engage in diverse conversations could enhance user satisfaction and make the application more comprehensive in meeting users' informational needs.

The third limitation was the diet plan presentation is not really in details. The diet plan generated by the application, while serving as a valuable tool for users, is presented in a text-based format that lacks comprehensive detail. This limitation can impact user understanding and adherence to the proposed diet plans.

The fourth limitation is the absence of notification features. A notable omission in the current iteration of the diet consultant application is the absence of notification features. Notifications can serve as valuable reminders for users to record progress or stay on track with their dietary goals. Incorporating a notification system could enhance user engagement and help individuals adhere to their diet plans more effectively. Implementing customizable notifications for various aspects of the diet and health journey could address this limitation.

The last limitation is the application language only restricted to English. An inherent limitation of the application is its language restriction to English. While this choice enables accessibility to a broad user base, it also poses barriers for non-English-speaking individuals who may benefit from the app's features. Expanding language support to cater to a more diverse audience could broaden the application's reach and impact, making it more inclusive and globally relevant.

## 8.4 Recommendation for Future Work

The Android-based diet consulting application still has room for improvement in light of the limitations mentioned above. Table 8.1 outlines recommendations for future enhancements. These suggestions can provide valuable guidance to future developers in enhancing the application's functionality and addressing its limitations, thereby improving the overall user experience.

Table 8.1: Recommendations

| Limitation | Recommendation |
|---|---|
| 1. Obtaining Raw Data for Lean Mass and Fat Mass | Explore partnerships with fitness device manufacturers or health data providers to offer more accurate and real-time measurements. Educate users on proper measurement techniques and compatible devices. |
| 2. Chatbot Versatility Beyond Diet and Health Topics | Expand the chatbot's knowledge base to respond to a wider range of inquiries. Implement natural language processing and integrate with external knowledge sources. |
| 3. Diet Plan Presentation and Detail | Redesign diet plans to include visuals, meal timing recommendations, portion sizes, and detailed explanations about food choices and their nutritional benefits. Offer customizable diet plan formats. |

| | |
|---|---|
| 4. Absence of Notification Features | Implement a robust notification system, allowing users to receive customizable notifications for meal reminders, progress tracking, and other relevant aspects of their health journey. Utilize push notifications, email reminders, and in-app alerts. |
| 5. Language Restriction to English | Introduce multilingual support within the application, allowing users to select their preferred language, including Malay, Mandarin, Cantonese, and others. Collaborate with professional translators or use machine translation tools. |

# REFERENCES


Anggraeni, N. (n.d.). Calorie Counter Information System Development for Weight Loss Program based on Android. [online] Available at: <https://www.scitepress.org/Papers/2018/99460/99460.pdf> [Accessed 1 Apr. 2023].

Anon (2019). *When It Comes to Health, Muscle Mass Matters*. [online] The Doctor Weighs In. Available at: <https://thedoctorweighsin.com/when-it-comes-to-health-muscle-mass-matters/> [Accessed 10 Apr. 2023].

Anon (2022). *10 best Android diet apps and Android nutrition apps*. [online] Android Authority. Available at: https://www.androidauthority.com/best-android-diet-apps-and-android-nutrition-apps-569702/#9 [Accessed 20 Apr. 2023].

Anon (n.d.). *What's the difference between a nutritionist and a dietitian?* [online] www.deakin.edu.au. Available at: https://www.deakin.edu.au/articles/study-areas/food-and-nutrition/whats-the-difference-between-a-nutritionist-and-a-dietitian [Accessed 20 Apr. 2023].

Anon. (2019). *SDLC - Iterative Incremental Model*. [online] Available at: <https://www.tutorialspoint.com/adaptive_software_development/sdlc_iterative_incremental_model.htm> [Accessed 12 Mar. 2023].

Farid (2021). DIETFIT MOBILE APPLICATION USING RULE-BASED SYSTEM TECHNIQUE MUHAMMAD FARID. [online] Available at: <https://myfik.unisza.edu.my/www/fyp/fyp20sem1/report/051138.pdf> [Accessed 3 Apr. 2023].

Forbes (n.d.). *Forbes Health - Your Health. Your Decisions. Our Support.* [online] Forbes Health. Available at: <https://www.forbes.com/health> [Accessed 10 Apr. 2023].

Gadam, S. (2023). *What is iterative and incremental development? Process, examples*. [online] LogRocket Blog. Available at: <https://blog.logrocket.com/product-management/what-is-iterative-incremental-development-process-examples/#what-does-an-iterative-and-incremental-development-process-look-like> [Accessed 20 Apr. 2023].

GeeksforGeeks (2017). *Decision Tree Introduction with example*. [online] GeeksforGeeks. Available at: <https://www.geeksforgeeks.org/decision-tree-introduction-example/.>

Global Nutrition Report. (n.d.). *Country Nutrition Profiles*. [online] Available at: <https://globalnutritionreport.org/resources/nutrition-profiles/asia/south-eastern-asia/malaysia/> [Accessed 13 Mar. 2023].

Health, S. (n.d.). *Eating Well for Mental Health | Sutter Health*. [online] www.sutterhealth.org. Available at: <https://www.sutterhealth.org/health/nutrition/eating-well-for-mental-health> [Accessed 12 Mar. 2023].

Kapil, A.R. (2022). *Advantages and disadvantages of decision tree in machine learning*. [online] Blogs & Updates on Data Science, Business Analytics, AI Machine Learning. Available at: <https://www.analytixlabs.co.in/blog/decision-tree-algorithm/.>

Kirsten Nunez (2019). *Muscle Mass Percentage Averages and How to Calculate It*. [online] Healthline. Available at: <https://www.healthline.com/health/muscle-mass-percentage#muscle-percentage.>

Kosarenko, Y. (2021). *How to Create Decision Trees for Business Rules Analysis*. [online] Why Change. Available at: <https://why-change.com/2021/11/13/how-to-create-decision-trees-for-business-rules-analysis/.>

Ministry of Health Malaysia (2020). *Buku_Nutrition_Research_Priorities_in_Malaysia(2021-2025)*. [online] www.moh.gov.my. Available at: <https://www.moh.gov.my/moh/resources/Pemakanan/Buku_Nutrition_Research_Priorities_in_Malaysia_(2021-2025).pdf> [Accessed 2 Apr. 2023].

Othman, A.A. (2021). *LETTER | Bad eating habits can lead to obesity*. [online] Malaysiakini. Available at: <https://www.malaysiakini.com/letters/577543> [Accessed 14 Mar. 2023].

Parisa Pouladzadeh (2012). A Novel SVM Based Food Recognition Method for Calorie Measurement Applications. [online] Available at: <https://www.researchgate.net/publication/261164203_A_Novel_SVM_Based_Food_Recognition_Method_for_Calorie_Measurement_Applications> [Accessed 3 Apr. 2023].

Raj, A. (2022). *Everything About Support Vector Classification — Above and Beyond*. [online] Medium. Available at: <https://towardsdatascience.com/everything-about-svm-classification-above-and-beyond-cc665bfd993e#:~:text=Disadvantages%20of%20SVM%20Classifier%3A&text=SVM%20algorithm%20is%20not%20suitable.>

Ramadas, A., Tham, S.M., Lalani, S.A. and Shyam, S. (2021). Diet Quality of Malaysians across Lifespan: A Scoping Review of Evidence in a Multi-Ethnic Population. *Nutrients*, 13(4), p.1380.

Raphael, J.R. (2020). *Android versions: A living history from 1.0 to 11*. [online] Computerworld. Available at: <https://www.computerworld.com/article/3235946/android-versions-a-living-history-from-1-0-to-today.html?page=2> [Accessed 11 Mar. 2023].

Salah Gad Foda (2012). *Chemical threats diagnosis expert system*. [online] https://www.researchgate.net/figure/Basic-architecture-of-a-rule-based-expert-system_fig3_236166495. Available at: <https://www.researchgate.net/figure/Basic-architecture-of-a-rule-based-expert-system_fig3_236166495> [Accessed 1 Apr. 2023].

Williams, T. (2021). Why is quantitative research important? [blog] *Grand Canyon University*. Available at: <https://www.gcu.edu/blog/doctoral-journey/why-quantitative-research-important> [Accessed 14 Mar. 2023].

Wittwer, J. (2018). *Printable BMI Chart*. [online] Vertex42.com. Available at: <https://www.vertex42.com/ExcelTemplates/bmi-chart.html.>

Wrike (n.d.). *What is Iterative and Incremental Development?* [online] Wrike. Available at: <https://www.wrike.com/project-management-guide/faq/what-is-iterative-incremental-development/.>

Zheng Yii, L. (2021). *Diet Link DEC Second-Edition 2021*. [online] https://www.dietitians.org.my/. Available at: <https://storage.unitedwebnetwork.com/files/290/5a3c3c0c5e3e0343f34b86adff566447.pdf >[Accessed 2 Apr. 2023].

# APPENDICES

## Appendix A: Complete Survey Form Questions

Figure A-1: Introduction Page of The Survey Form

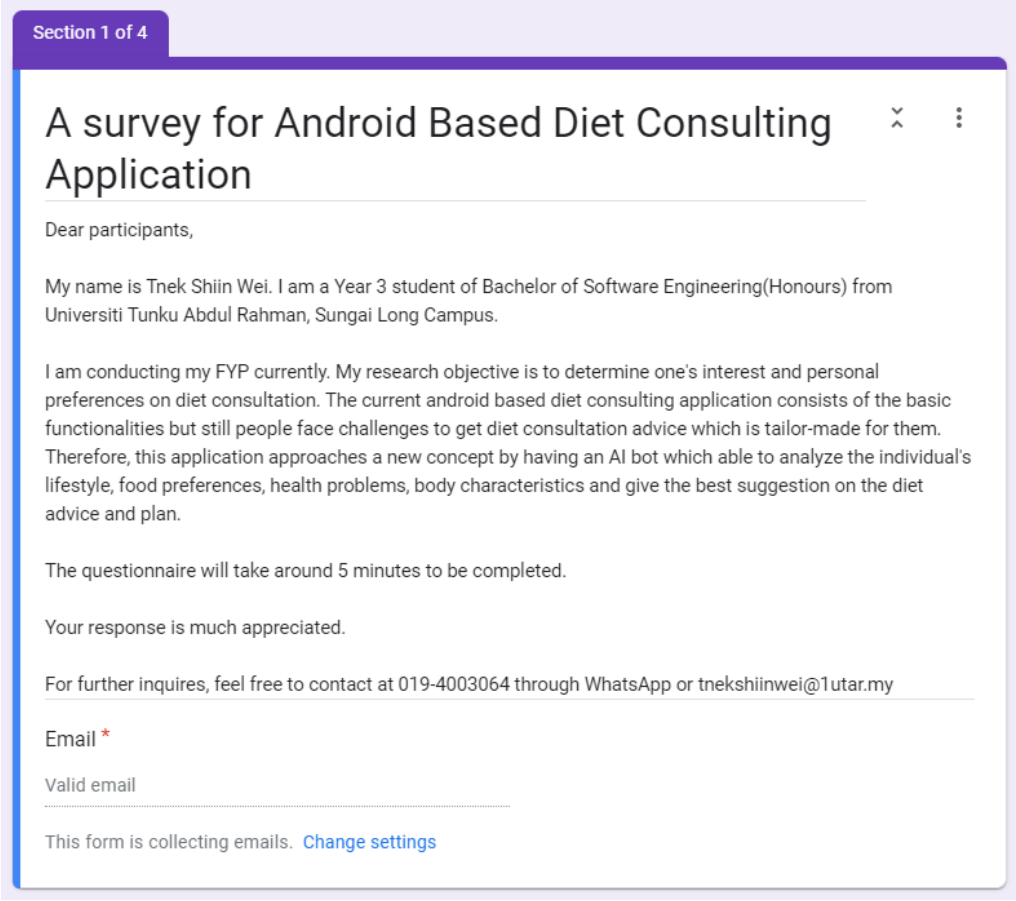


Figure A-2: Demographic Section of the Survey Form

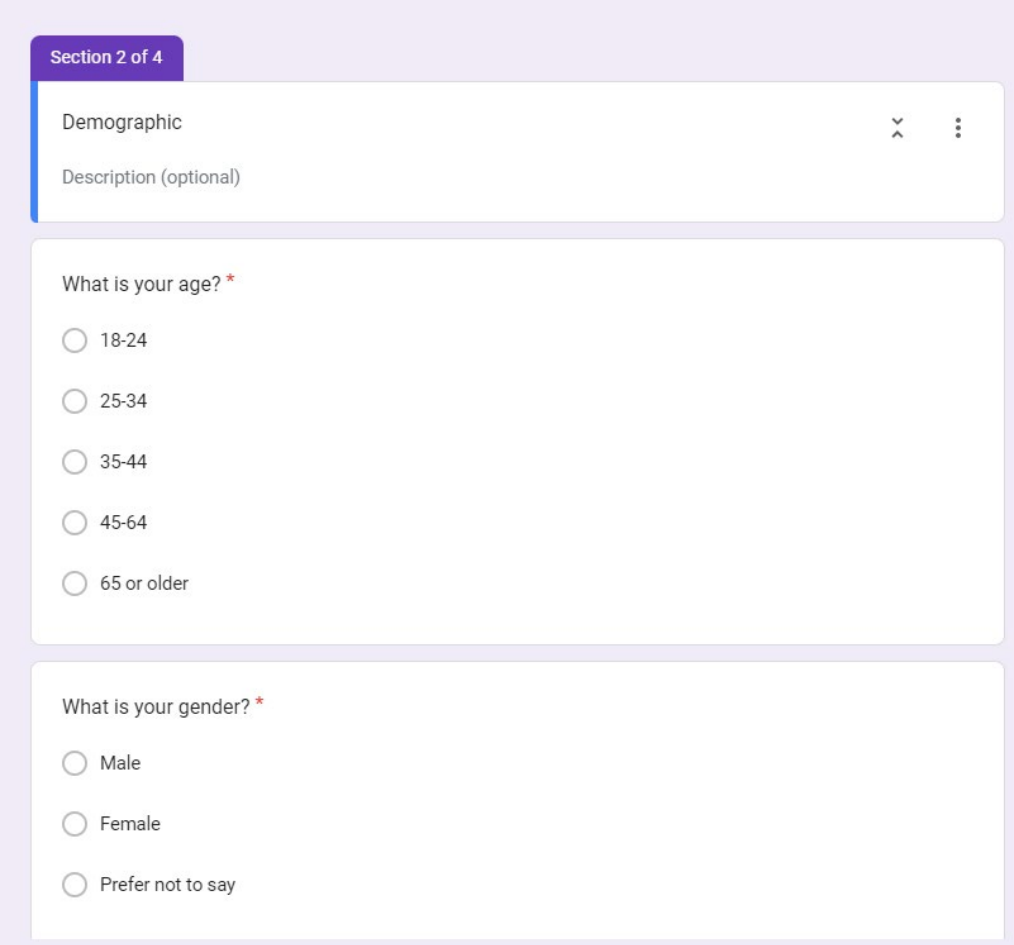


Figure A-3: Section 3 of the Survery form -1

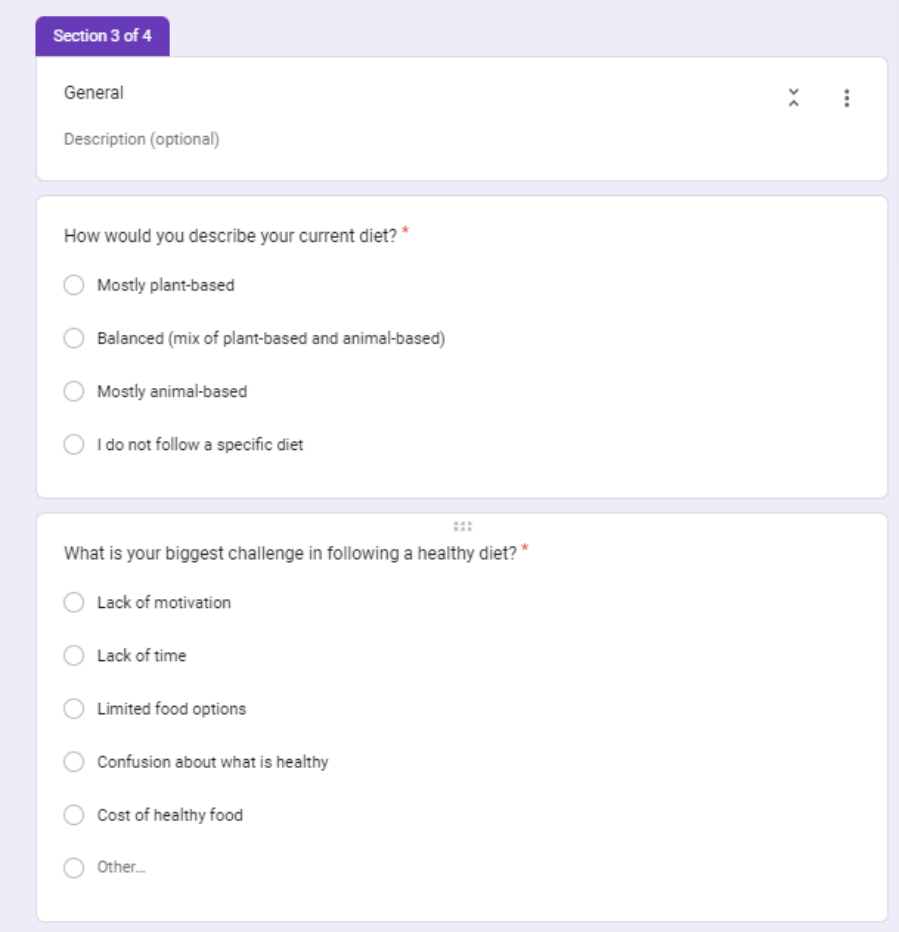


Figure A-4: Section 3 of the Survery form -2

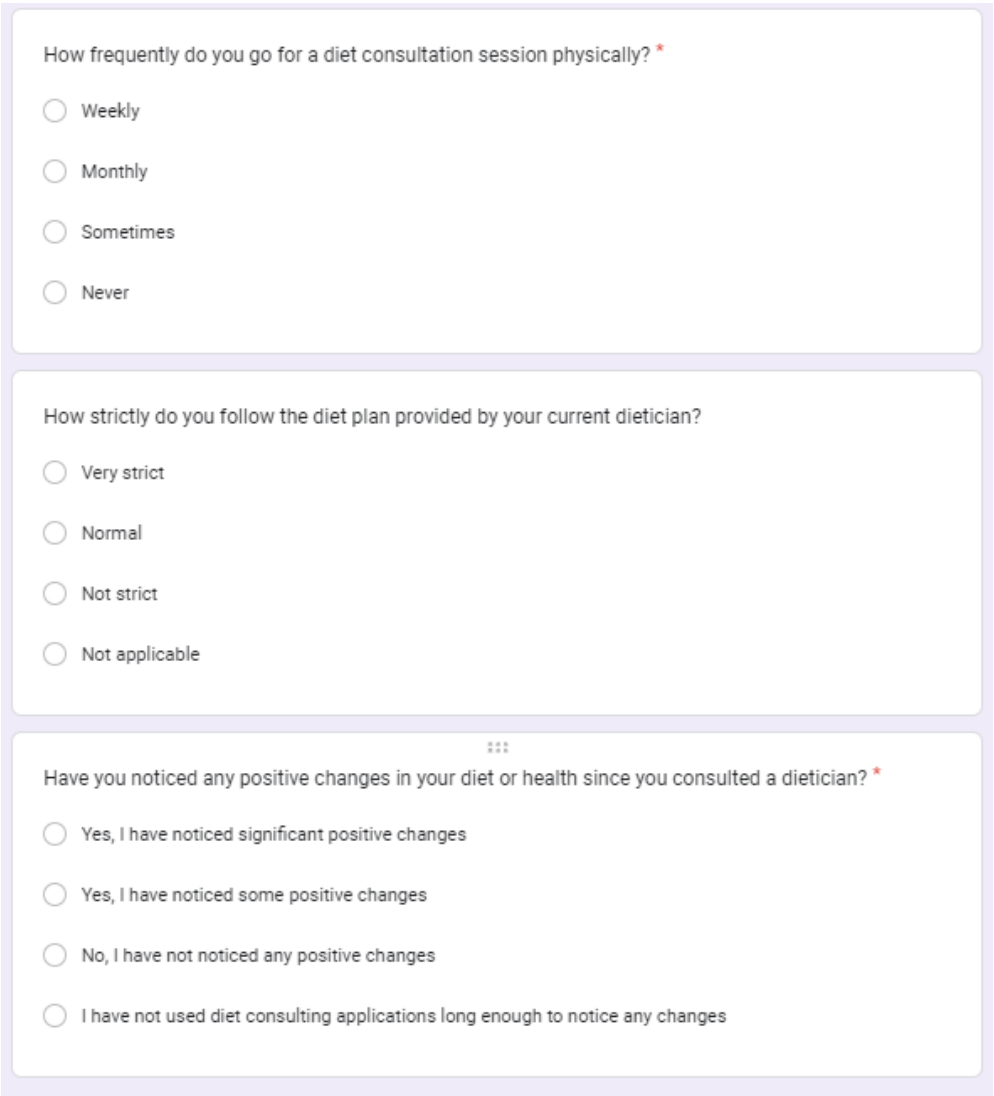


Figure A-5: Final Section of the Survey Form-1

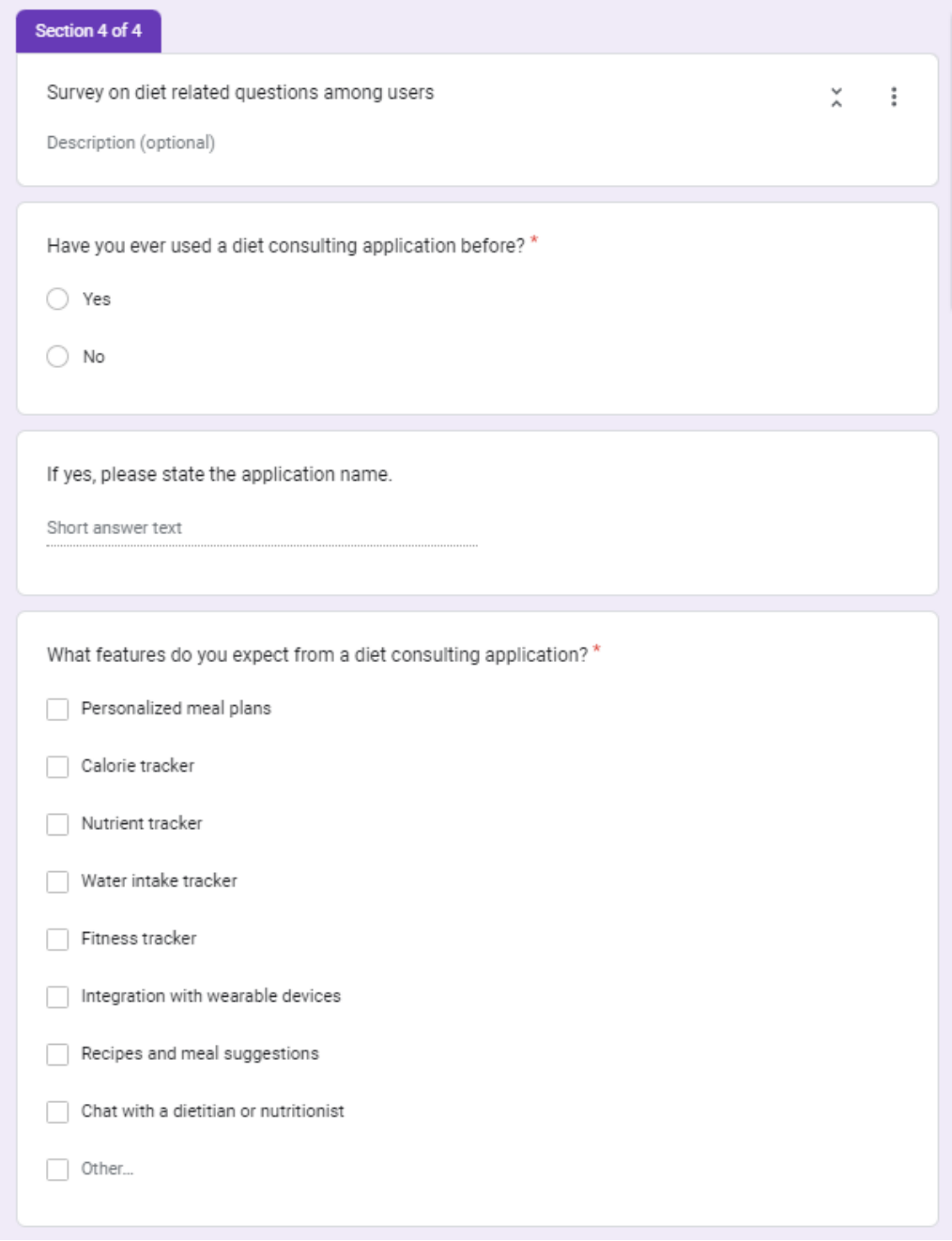


Figure A-6 Final Section of the Survey Form-2

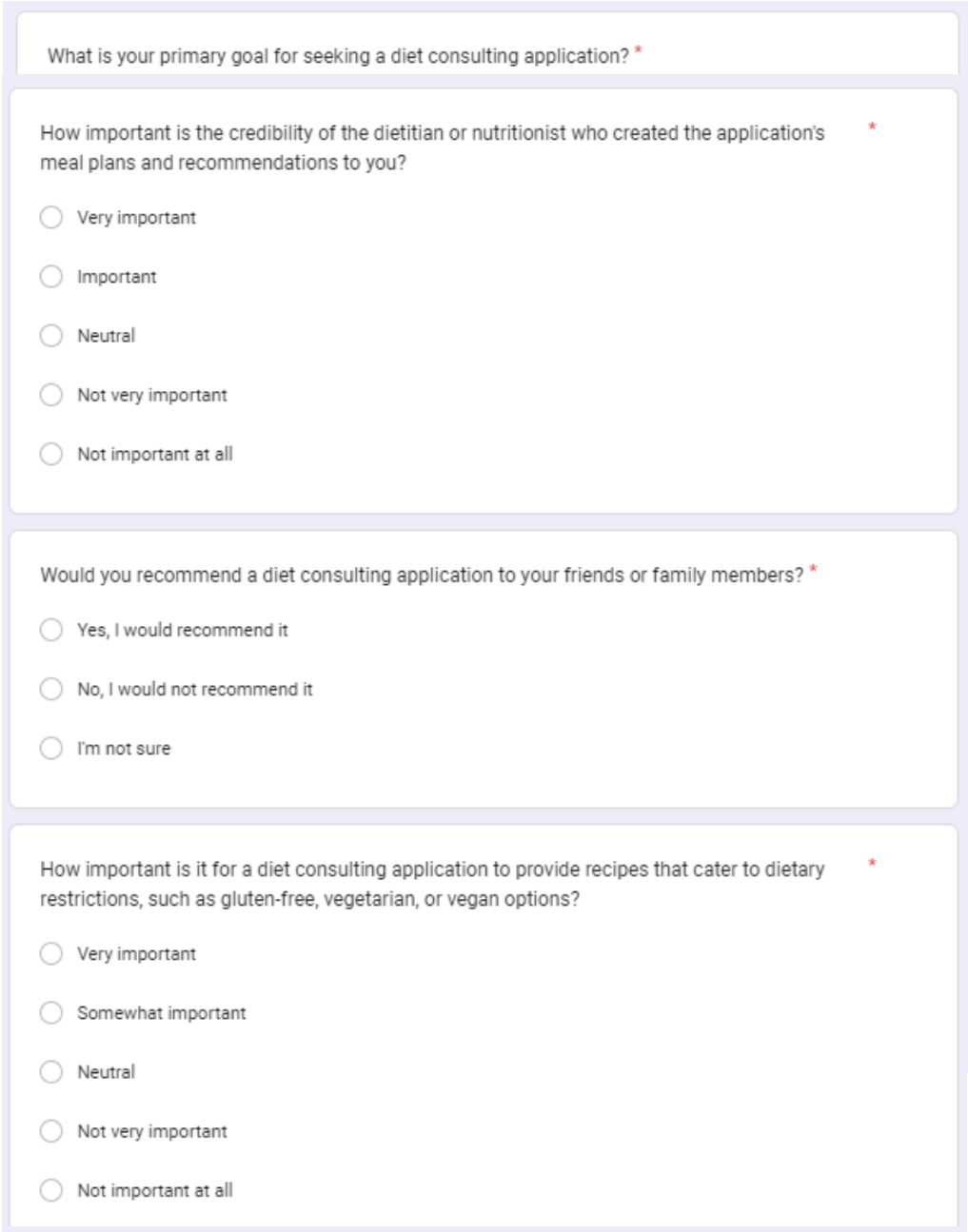
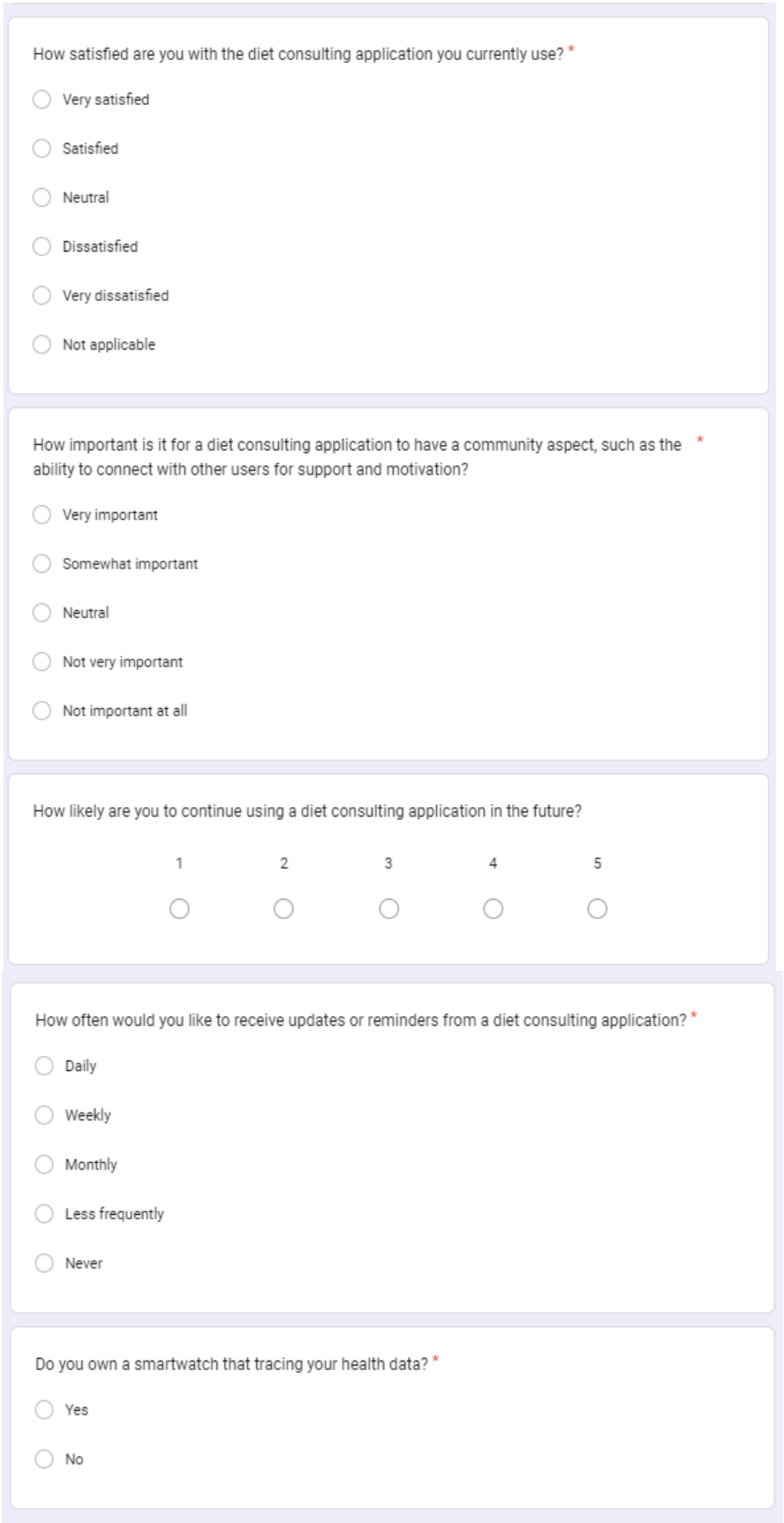


Figure A-7: Final Section of the Survey Form-3



Appendix B: Unit Testing

| **Test Case ID** | UTC001 | **Test Name** | Test login with correct credential | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the system will show login success | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User enters the valid and correct email and password.<br>2. User presses the login button | | | | Email: tnekshiinwei@gmail.com<br>Password: 123123 | |
| **Expected Result** | | | | **Actual Result** | |
| The system redirects to progress screen. | | | | The system redirects to progress screen | |

| **Test Case ID** | UTC002 | **Test Name** | Test login with correct email but invalid password | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the system will show error message | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User enters correct email but invalid password.<br>2. User presses the login button. | | | | Email: tnekshiinwei@gmail.com<br>Password: 123111 | |
| **Expected Result** | | | | **Actual Result** | |
| "Wrong Password" | | | | "Wrong Password" | |

| **Test Case ID** | UTC003 | **Test Name** | Test login with invalid email | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the system will show error message | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User enters invalid email and password.<br>2. User presses on the login button. | | | | Email: tnekshiinwei1@gmail.com | |
| **Expected Result** | | | | **Actual Result** | |
| "No user corresponding to the given email address" | | | | "No user corresponding to the given email address" | |

| **Test Case ID** | UTC004 | **Test Name** | Test login with an empty email | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User enters empty email.<br>2. User presses on the login button. | | | | Email: “” | |
| **Expected Result** | | | | **Actual Result** | |
| “Enter valid email” | | | | “Enter valid email” | |

| **Test Case ID** | UTC005 | **Test Name** | Test login with an empty password | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the system will show error message | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. Enter enters email and leaves password empty.<br>2. User presses on the login button. | | | | Email: tnekshiinwei@gmail.com<br>Password: “” | |
| **Expected Result** | | | | **Actual Result** | |
| “Password must be more than 5 characters” | | | | “Password must be more than 5 characters” | |

| **Test Case ID** | UTC006 | **Test Name** | Test sign up with valid information | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the system will show sign up success | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User enters the valid first name, last name, age, weight, height, gender, medical history, food preferences, email, password and confirm password.<br>2. User presses on agree the Terms of Use.<br>3. User presses on “Sign Up” button | | | | firstName: “Shiin”<br>lastName: “Wei”<br>Age: 23<br>Weight: 60<br>Height: 173<br>Gender: “Male”<br>Medical History: “None”<br>Food Preferences: “None”<br>Email: “tnekshiinwei@gmail.com”<br>Password: “123123”<br>Confirm Password: “123123” | |
| **Expected Result** | | | | **Actual Result** | |
| User is redirected to Progress Screen | | | | User is redirected to Progress Screen. | |

| **Test Case ID** | UTC007 | **Test Name** | Test sign up with all empty fields | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the system will show error message | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User presses on agree the Terms of Use.<br>2. User presses on "Sign Up" button | | | | | |
| **Expected Result** | | | | **Actual Result** | |
| "Please fill required field" | | | | "Please fill required field" | |

| **Test Case ID** | UTC008 | **Test Name** | Test sign up with valid information but with empty first name and last name | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the system will show error message | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User enters only age, weight, height, gender, medical history, food preferences, email, password and confirm password.<br>2. User presses on agree the Terms of Use.<br>3. User presses on "Sign Up" button | | | | Age: 23<br>Weight: 60<br>Height: 173<br>Gender: "Male"<br>Medical History: "None"<br>Food Preferences: "None"<br>Email: "tnekshiinwei@gmail.com"<br>Password: "123123"<br>Confirm Password: "123123" | |
| **Expected Result** | | | | **Actual Result** | |
| "Name is required" | | | | "Name is required" | |

| **Test Case ID** | UTC009 | **Test Name** | Test sign up with valid information but with empty email | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the system will show error message | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User enters the valid first name, last name, age, weight, height, gender, medical | | | | firstName: "Shiin"<br>lastName: "Wei"<br>Age: 23 | |

| | |
|---|---|
| history, food preferences, password and confirm password.<br>2. User presses on agree the Terms of Use.<br>User presses on “Sign Up” button | Weight: 60<br>Height: 173<br>Gender: “Male”<br>Medical History: “None”<br>Food Preferences: “None”<br>Password: “123123”<br>Confirm Password: “123123” |
| **Expected Result** | **Actual Result** |
| “Enter Valid Email” | “Enter Valid Email” |

| | | | | | |
|---|---|---|---|---|---|
| **Test Case ID** | UTC010 | **Test Name** | Test sign up with valid information but with invalid email | **Test Status** | Pass |
| **Test Case Description** | | Examine whether the system will show error message | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User enters the valid first name, last name, age, weight, height, gender, medical history, food preferences, email, password and confirm password.<br>2. User presses on agree the Terms of Use.<br>3. User presses on “Sign Up” button | | | | firstName: “Shiin”<br>lastName: “Wei”<br>Age: 23<br>Weight: 60<br>Height: 173<br>Gender: “Male”<br>Medical History: “None”<br>Food Preferences: “None”<br>Email: “tnekshiinwei”<br>Password: “123123”<br>Confirm Password: “123123” | |
| **Expected Result** | | | | **Actual Result** | |
| “Enter Valid Email” | | | | “Enter Valid Email” | |

| | | | | | |
|---|---|---|---|---|---|
| **Test Case ID** | UTC011 | **Test Name** | Test sign up with valid information but with empty password | **Test Status** | Pass |
| **Test Case Description** | | Examine whether the system will show error message | | | |

| **Test Case Scenario** | **Test Data** |
|---|---|
| 1. User enters the valid first name, last name, age, weight, height, gender, medical history, food preferences, email.<br>2. User presses on agree the Terms of Use.<br>3. User presses on "Sign Up" button | firstName: "Shiin"<br>lastName: "Wei"<br>Age: 23<br>Weight: 60<br>Height: 173<br>Gender: "Male"<br>Medical History: "None"<br>Food Preferences: "None"<br>Email: "tnekshiinwei@gmail.com" |
| **Expected Result** | **Actual Result** |
| "Password must be more than 5 characters" | "Password must be more than 5 characters" |

| **Test Case ID** | UTC012 | **Test Name** | Test sign up with valid information but with empty confirm password | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | Examine whether the system will show error message | | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User enters the valid first name, last name, age, weight, height, gender, medical history, food preferences, email, password.<br>2. User presses on agree the Terms of Use.<br>3. User presses on "Sign Up" button | | | | firstName: "Shiin"<br>lastName: "Wei"<br>Age: 23<br>Weight: 60<br>Height: 173<br>Gender: "Male"<br>Medical History: "None"<br>Food Preferences: "None"<br>Email: "tnekshiinwei@gmail.com"<br>Password: "123123" | |
| **Expected Result** | | | | **Actual Result** | |
| "Confirm password is required" | | | | "Confirm password is required" | |

| **Test Case ID** | UTC013 | **Test Name** | Test user signs out current user | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the system will end the current user's session | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User presses on the logout button | | | | | |

| **Expected Result** | **Actual Result** |
|---|---|
| User is redirected to the onboarding screen | User is redirected to the onboarding screen |

| **Test Case ID** | UTC014 | **Test Name** | Test get current user's profile information | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | Examine whether the system able to retrieve user's profile information | | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User presses on the profile button | | | | | |
| **Expected Result** | | | | **Actual Result** | |
| User is redirected to the profile screen with the full list of user profile information | | | | User is redirected to the profile screen with the full list of user profile information | |

| **Test Case ID** | UTC015 | **Test Name** | Test modify user's profile | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | Examine whether the system able to update user's profile information | | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User presses on the edit icon button<br>2. User chooses the specific information to edit and enter valid latest information.<br>3. Then user presses on the save button. | | | | Weight: 60 | |
| **Expected Result** | | | | **Actual Result** | |
| The latest information is updated in the database and interface. | | | | The latest information is updated in the database and interface. | |

| **Test Case ID** | UTC016 | **Test Name** | Test modify user's profile using invalid input | **Test Status** | Fail |
|---|---|---|---|---|---|
| **Test Case Description** | Examine whether the system able to show error message | | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User presses on the edit icon button<br>2. User chooses the specific information to edit and enter valid latest information. | | | | Weight: "L" | |

| | |
|---|---|
| 3. Then user presses on the save button. | |
| **Expected Result** | **Actual Result** |
| "Please enter a valid input" | Weight value is updated with initial value. |

| | | | | | |
|---|---|---|---|---|---|
| **Test Case ID** | UTC017 | **Test Name** | Test get user progress information | **Test Status** | Pass |
| **Test Case Description** | | Examine whether the system able to retrieve user's progress information | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User is redirected to the progress screen | | | | | |
| **Expected Result** | | | | **Actual Result** | |
| User is shown with the progress record data | | | | User is shown with the progress record data | |

| | | | | | |
|---|---|---|---|---|---|
| **Test Case ID** | UTC018 | **Test Name** | Test get user progress information if no progress information found. | **Test Status** | Pass |
| **Test Case Description** | | Examine whether the system able to show error message | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User is redirected to the progress screen | | | | | |
| **Expected Result** | | | | **Actual Result** | |
| "No progress data found" | | | | "No progress data found" | |

| | | | | | |
|---|---|---|---|---|---|
| **Test Case ID** | UTC019 | **Test Name** | Test add user progress | **Test Status** | Pass |
| **Test Case Description** | | Examine whether the system able to retrieve user's latest progress information | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User presses on the add floating action button.<br>2. User input their fat mass and lean mass value.<br>3. User presses on save button. | | | | Fat mass: 20<br>Lean mass: 40 | |
| **Expected Result** | | | | **Actual Result** | |
| The user latest recorded progress information is shown in the progress screen. | | | | The user latest recorded progress | |

| | |
|---|---|
| | information is shown in the progress screen. |

| **Test Case ID** | UTC020 | **Test Name** | Test add user progress with invalid value | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the system able to show error message | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User presses on the add floating action button.<br>2. User input invalid fat mass and lean mass value.<br>3. User presses on save button. | | | | Fat mass: 1<br>Lean mass: 20 | |
| **Expected Result** | | | | **Actual Result** | |
| Please enter a value between 5 and 100. | | | | Please enter a value between 5 and 100. | |

| **Test Case ID** | UTC021 | **Test Name** | Test view specific existing diet plan | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the system able to retrieve user's specific existing diet plan information | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User navigates to view diet plan screen.<br>2. User presses to view the specific diet plan. | | | | | |
| **Expected Result** | | | | **Actual Result** | |
| The diet plan selected information is shown to the user | | | | The diet plan selected information is shown to the user | |

| **Test Case ID** | UTC022 | **Test Name** | Test delete specific existing diet plan | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the system able to delete the chosen existing diet plan. | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User navigates to view diet plan screen.<br>2. User presses to view the specific diet plan.<br>3. User presses the delete icon button. | | | | | |
| **Expected Result** | | | | **Actual Result** | |
| The selected diet plan is deleted from the firestore. | | | | The selected diet plan is deleted from the firestore. | |

| **Test Case ID** | UTC023 | **Test Name** | Test generate diet plan | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the system able to prompt the diet plan information and update the latest diet plan to the database. | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User navigates to view diet plan screen.<br>2. User chooses the goal preferences.<br>3. User presses on the get diet plan button | | | | Goal Preferences: "Losing Weight" | |
| **Expected Result** | | | | **Actual Result** | |
| The system will generate the diet plan. | | | | The system will generate the diet plan. | |

| **Test Case ID** | UTC024 | **Test Name** | Test generate diet plan without an existing progress data | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the system able to show message | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User navigates to view diet plan screen.<br>2. User chooses the goal preferences. | | | | Goal Preferences: "Losing Weight" | |
| **Expected Result** | | | | **Actual Result** | |
| "No progress data detected" | | | | "No progress data detected" | |

| **Test Case ID** | UTC025 | **Test Name** | Test record feedback | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the system able to retrieve user's feedback | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User navigates to feedback screen<br>2. User fills in the feedback form.<br>3. User presses on the submit button | | | | Name: "Shiin Wei"<br>Email: "tnekshiinwei@gmail.com<br>Subject: "Testing"<br>Comment: "Testing ya" | |
| **Expected Result** | | | | **Actual Result** | |
| "Feedback submitted successfully" | | | | "Feedback submitted successfully" | |

| **Test Case ID** | UTC026 | **Test Name** | Test record feedback | **Test Status** | Pass |
|---|---|---|---|---|---|

| | | | | | |
|---|---|---|---|---|---|
| | | | with blank input | | |
| **Test Case Description** | | Examine whether the system able to show message | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User navigates to feedback screen<br>2. User presses on the submit button | | | | | |
| **Expected Result** | | | | **Actual Result** | |
| "Please enter your name"<br>"Please enter your email"<br>"Please enter your subject"<br>"Please enter your comment" | | | | "Please enter your name"<br>"Please enter your email"<br>"Please enter your subject"<br>"Please enter your comment" | |

| **Test Case ID** | UTC027 | **Test Name** | Test diet chatbot with a valid question | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the system able to prompt response according to user input | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User navigates to chatbot screen<br>2. User input their questions and presses on the send button. | | | | Message: "Please help me on my diet" | |
| **Expected Result** | | | | **Actual Result** | |
| The chatbot will give response according to the message content. | | | | The chatbot will give response according to the message content. | |

| **Test Case ID** | UTC028 | **Test Name** | Test diet chatbot with a blank input | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the system able to show error message | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User navigates to chatbot screen<br>2. User presses on the send button. | | | | | |
| **Expected Result** | | | | **Actual Result** | |
| "Please enter a valid question." | | | | "Please enter a valid question." | |

Appendix C: Integration Testing

| **Test Case ID** | ITC001 | **Test Name** | Test login screen | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the user can navigate to login page | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User presses on the login button in the onboarding screen | | | | - | |
| **Expected Result** | | | | **Actual Result** | |
| User is able to access the login page | | | | User is able to access the login page | |

| **Test Case ID** | ITC002 | **Test Name** | Test progress screen if user is authenticated | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the user can navigate to progress page if user is authenticated. | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User presses on the progress screen in the bottom navigation bar. | | | | - | |
| **Expected Result** | | | | **Actual Result** | |
| User will be redirected to the progress screen | | | | User is redirected to the progress screen | |

| **Test Case ID** | ITC003 | **Test Name** | Test progress screen if user is not authenticated | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the user is redirect back to login page if the user is not authenticated. | | | |
| **Test Case Scenario** | | | | **Test Data** | |

| | |
|---|---|
| 1. User pressed on the login button after filling the invalid credential information. | - |
| **Expected Result** | **Actual Result** |
| User will be redirected back to login page. | User will be redirected back to login page. |

| | | | | | |
|---|---|---|---|---|---|
| **Test Case ID** | ITC004 | **Test Name** | Test log out | **Test Status** | Pass |
| **Test Case Description** | | Examine whether the user is removed from session and redirect to the onboarding screen. | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User presses on the log out button. | | | | - | |
| **Expected Result** | | | | **Actual Result** | |
| User will be redirected back to the onboarding screen. | | | | User will be redirected back to the onboarding screen. | |

| | | | | | |
|---|---|---|---|---|---|
| **Test Case ID** | ITC005 | **Test Name** | Test sign up screen | **Test Status** | Pass |
| **Test Case Description** | | Examine whether user can navigate to sign up screen. | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User presses on the sign-up button in the onboarding screen. | | | | - | |
| **Expected Result** | | | | **Actual Result** | |
| User will be navigated to the sign up screen. | | | | User will be navigated to the sign up screen. | |

| | | | | | |
|---|---|---|---|---|---|
| **Test Case ID** | ITC006 | **Test Name** | Test submit sign up | **Test Status** | Pass |
| **Test Case Description** | | Examine whether user can navigate to user's progress screen | | | |

| **Test Case Scenario** | **Test Data** |
|---|---|
| 1. User fills all the information in the sign up screen.<br>2. User presses on the sign up button | - |
| **Expected Result** | **Actual Result** |
| User is navigated to the user's progress screen. | User is navigated to the user's progress screen. |

| **Test Case ID** | ITC007 | **Test Name** | Test user profile screen | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | Examine whether the user can navigate to user profile screen after the user is authenticated. | | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User presses on the profile button | | | | | |
| **Expected Result** | | | | **Actual Result** | |
| User is redirected to the profile screen. | | | | User is redirected to the profile screen. | |

| **Test Case ID** | ITC008 | **Test Name** | Test update profile information | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | Examine whether the new information is updated to the backend | | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User fills in the information in the profile screen. | | | | | |
| **Expected Result** | | | | **Actual Result** | |
| The firestore database is updated with the latest profile data. | | | | The firestore database is updated with the latest profile data. | |

| **Test Case ID** | ITC009 | **Test Name** | Test view diet plan screen | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the user can navigate to view diet plan screen after the user is authenticated. | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User presses on the view diet plan in the bottom navigation bar. | | | | | |
| **Expected Result** | | | | **Actual Result** | |
| User is redirected to the view diet plan page. | | | | User is redirected to the view diet plan page. | |

| **Test Case ID** | ITC010 | **Test Name** | Test generate diet plan screen | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the user can navigate to generate diet plan screen after pressing on “Generate” button. | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User chooses their goal preferences and presses on the generate button. | | | | | |
| **Expected Result** | | | | **Actual Result** | |
| User is redirected to the generate diet plan screen. | | | | User is redirected to the generate diet plan screen. | |

| **Test Case ID** | ITC011 | **Test Name** | Test add diet plan | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the diet plan information is added to database | | | |
| **Test Case Scenario** | | | | **Test Data** | |

| | |
|---|---|
| 1. User presses on the get diet plan button in the generate diet plan screen. | - |
| **Expected Result** | **Actual Result** |
| The latest diet plan generated is added to the Firestore Database. | The latest diet plan generated is added to the Firestore Database. |

| | | | | | |
|---|---|---|---|---|---|
| **Test Case ID** | ITC012 | **Test Name** | Test delete diet plan | **Test Status** | Pass |
| **Test Case Description** | | Examine whether the diet plan information is deleted from database | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User presses on the existing diet plan listed.<br>2. User presses on the delete icon of a specific diet plan. | | | | - | |
| **Expected Result** | | | | **Actual Result** | |
| The diet plan selected is deleted from the Firestore Database. | | | | The diet plan selected is deleted from the Firestore Database. | |

| | | | | | |
|---|---|---|---|---|---|
| **Test Case ID** | ITC013 | **Test Name** | Test add progress | **Test Status** | Pass |
| **Test Case Description** | | Examine whether the add progress information is added to the backend | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User presses on the add floating action button in progress screen.<br>2. User fills in the required fields. | | | | - | |
| **Expected Result** | | | | **Actual Result** | |
| The latest progress data is added to the Firestore Database. | | | | The latest progress data is added to the Firestore Database. | |

| **Test Case ID** | ITC014 | **Test Name** | Test feedback screen | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether user can navigate to feedback screen after user is authenticated | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User presses on the feedback button in the bottom navigation bar. | | | | - | |
| **Expected Result** | | | | **Actual Result** | |
| User is navigated to feedback screen. | | | | User is navigated to feedback screen. | |

| **Test Case ID** | ITC015 | **Test Name** | Test add feedback | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether the feedback information is added to the back end | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User fills in the required information in the feedback screen. | | | | - | |
| **Expected Result** | | | | **Actual Result** | |
| The feedback information submitted is added to the Firestore Database. | | | | The feedback information submitted is added to the Firestore Database. | |

| **Test Case ID** | ITC016 | **Test Name** | Test chatbot screen | **Test Status** | Pass |
|---|---|---|---|---|---|
| **Test Case Description** | | Examine whether user can navigate to chatbot after the user is authenticated | | | |
| **Test Case Scenario** | | | | **Test Data** | |
| 1. User presses on the chatbot button in the bottom nagivation bar. | | | | - | |

| **Expected Result** | **Actual Result** |
|---|---|
| User is navigated to the chatbot screen. | User is navigated to the chatbot screen. |

Appendix D: User Acceptance Test

| **Test Case ID** | UAT001 | **Test Name** | Test login with correct credential |
|---|---|---|---|
| **Test Case Description** | | Examine whether the system will show login success | |
| **Test Case Scenario** | | | **Test Data** |
| 1. User enters the valid and correct email and password.<br>2. User presses the login button | | | Email: tnekshiinwei@gmail.com<br>Password: 123123 |
| **Expected Result** | | | **Actual Result** |
| The system redirects to progress screen. | | | The system redirects to progress screen. |
| **Test Status** | | | **Comments** |
| 1. Pass<br>2. Pass<br>3. Pass<br>4. Pass<br>5. Pass | | | |

| **Test Case ID** | UAT002 | **Test Name** | Test login with invalid credential |
|---|---|---|---|
| **Test Case Description** | | Examine whether the system will show error message | |
| **Test Case Scenario** | | | **Test Data** |
| 1. User enters correct email but invalid password.<br>2. User presses the login button. | | | Email: tnekshiinwei@gmail.com<br>Password: 123111 |
| **Expected Result** | | | **Actual Result** |

| | |
|---|---|
| "Wrong Password" | "Wrong Password" |
| **Test Status** | **Comments** |
| 1. Pass<br>2. Pass<br>3. Pass<br>4. Pass<br>5. Pass | |

| | | | |
|---|---|---|---|
| **Test Case ID** | UAT003 | **Test Name** | Test sign up with valid information |
| **Test Case Description** | | Examine whether the system will show sign up success | |
| **Test Case Scenario** | | | **Test Data** |
| 1. User enters the valid first name, last name, age, weight, height, gender, medical history, food preferences, email, password and confirm password.<br>2. User presses on agree the Terms of Use.<br>3. User presses on "Sign Up" button | | | firstName: "Shiin"<br>lastName: "Wei"<br>Age: 23<br>Weight: 60<br>Height: 173<br>Gender: "Male"<br>Medical History: "None"<br>Food Preferences: "None"<br>Email: "tnekshiinwei@gmail.com"<br>Password: "123123"<br>Confirm Password: "123123" |
| **Expected Result** | | | **Actual Result** |
| User is redirected to Progress Screen | | | User is redirected to Progress Screen |
| **Test Status** | | | **Comments** |
| 1. Pass<br>2. Pass | | | |

| | |
|---|---|
| 3. Pass<br>4. Pass<br>5. Pass | |

| **Test Case ID** | UAT004 | **Test Name** | Test sign up with invalid input |
|---|---|---|---|
| **Test Case Description** | | Examine whether the system will show error message | |
| **Test Case Scenario** | | | **Test Data** |
| 1. User enters the valid first name, last name, age, weight, height, gender, medical history, food preferences, password and confirm password.<br>2. User presses on agree the Terms of Use.<br>3. User presses on "Sign Up" button | | | firstName: "Shiin"<br>lastName: "Wei"<br>Age: 23<br>Weight: 60<br>Height: 173<br>Gender: "Male"<br>Medical History: "None"<br>Food Preferences: "None"<br>Email: ""<br>Password: "123123"<br>Confirm Password: "123123" |
| **Expected Result** | | | **Actual Result** |
| "Enter Valid Email" | | | "Enter Valid Email" |
| **Test Status** | | | **Comments** |
| 1. Pass<br>2. Pass<br>3. Pass<br>4. Pass<br>5. Pass | | | |

| **Test Case ID** | UAT005 | **Test Name** | Test use signs out current user |
|---|---|---|---|
| **Test Case Description** | | Examine whether the system will end the current user's session | |
| **Test Case Scenario** | | | **Test Data** |
| User presses on the logout button | | | |
| **Expected Result** | | | **Actual Result** |
| User is redirected to the onboarding screen | | | User is redirected to the onboarding screen |
| **Test Status** | | | **Comments** |
| 1. Pass<br>2. Pass<br>3. Pass<br>4. Pass<br>5. Pass | | | |

| **Test Case ID** | UAT006 | **Test Name** | Test get current user's profile information |
|---|---|---|---|
| **Test Case Description** | | Examine whether the system able to retrieve user's profile information | |
| **Test Case Scenario** | | | **Test Data** |
| 1. User presses on the profile button | | | - |
| **Expected Result** | | | **Actual Result** |
| User is redirected to the profile screen with the full list of user profile information | | | User is redirected to the profile screen with the full list of user profile information |
| **Test Status** | | | **Comments** |
| 1. Pass<br>2. Pass<br>3. Pass<br>4. Pass<br>5. Pass | | | |

| **Test Case ID** | UAT007 | **Test Name** | Test modify user's profile |
|---|---|---|---|
| **Test Case Description** | | Examine whether the system able to update user's profile information | |
| **Test Case Scenario** | | | **Test Data** |
| 1. User presses on the edit icon button<br>2. User chooses the specific information to edit and enter valid latest information.<br>3. Then user presses on the save button. | | | Weight: 60 |
| **Expected Result** | | | **Actual Result** |
| The latest information is updated in the database and interface. | | | The latest information is updated in the database and interface. |
| **Test Status** | | | **Comments** |
| 1. Pass<br>2. Pass<br>3. Pass<br>4. Pass<br>5. Pass | | | |

| **Test Case ID** | UAT008 | **Test Name** | Test modify user's profile using invalid input |
|---|---|---|---|
| **Test Case Description** | | Examine whether the system able to show error message | |
| **Test Case Scenario** | | | **Test Data** |
| 1. User presses on the edit icon button<br>2. User chooses the specific information to edit and enter valid latest information.<br>3. Then user presses on the save button. | | | Weight: "L" |
| **Expected Result** | | | **Actual Result** |

| | |
|---|---|
| "Please enter a valid input" | Weight value is updated with initial value. |
| **Test Status** | **Comments** |
| 1. Pass<br>2. Pass<br>3. Fail<br>4. Pass<br>5. Fail | Put some error handling event to handle this problem. |

| | | | |
|---|---|---|---|
| **Test Case ID** | UAT009 | **Test Name** | Test get user progress information |
| **Test Case Description** | | Examine whether the system able to retrieve user's progress information. | |
| **Test Case Scenario** | | | **Test Data** |
| 1. User is redirected to the progress screen | | | |
| **Expected Result** | | | **Actual Result** |
| User is shown with the progress record data | | | User is shown with the progress record data |
| **Test Status** | | | **Comments** |
| 1. Pass<br>2. Pass<br>3. Pass<br>4. Pass<br>5. Pass | | | |

| | | | |
|---|---|---|---|
| **Test Case ID** | UAT010 | **Test Name** | Test get user progress information if no progress information found |
| **Test Case Description** | | Examine whether the system able to show error message | |
| **Test Case Scenario** | | | **Test Data** |

| | |
|---|---|
| 1. User is redirected to the progress screen | |
| **Expected Result** | **Actual Result** |
| "No progress data found" | "No progress data found" |
| **Test Status** | **Comments** |
| 1. Pass<br>2. Pass<br>3. Pass<br>4. Pass<br>5. Pass | |

| | | | |
|---|---|---|---|
| **Test Case ID** | UAT011 | **Test Name** | Test add user progress |
| **Test Case Description** | | Examine whether the system able to retrieve user's latest progress information | |
| **Test Case Scenario** | | | **Test Data** |
| 1. User presses on the add floating action button.<br>2. User input their fat mass and lean mass value.<br>3. User presses on save button. | | | Fat mass: 20<br>Lean mass: 40 |
| **Expected Result** | | | **Actual Result** |
| The user latest recorded progress information is shown in the progress screen. | | | The user latest recorded progress information is shown in the progress screen. |
| **Test Status** | | | **Comments** |
| 1. Pass<br>2. Pass<br>3. Pass<br>4. Pass<br>5. Pass | | | |

| **Test Case ID** | UAT012 | **Test Name** | Test add user progress with invalid value |
|---|---|---|---|
| **Test Case Description** | | Examine whether the system able to show error message | |
| **Test Case Scenario** | | | **Test Data** |
| 1. User presses on the add floating action button.<br>2. User input invalid fat mass and lean mass value.<br>3. User presses on save button. | | | Fat mass: 1<br>Lean mass: 20 |
| **Expected Result** | | | **Actual Result** |
| Please enter a value between 5 and 100. | | | Please enter a value between 5 and 100. |
| **Test Status** | | | **Comments** |
| 1. Pass<br>2. Pass<br>3. Pass<br>4. Pass<br>5. Pass | | | |

| **Test Case ID** | UAT013 | **Test Name** | Test view specific existing diet plan |
|---|---|---|---|
| **Test Case Description** | | Examine whether the system able to retrieve user's specific existing diet plan information | |
| **Test Case Scenario** | | | **Test Data** |
| 1. User navigates to view diet plan screen.<br>2. User presses to view the specific diet plan. | | | |
| **Expected Result** | | | **Actual Result** |
| The diet plan selected information is shown to the user | | | The diet plan selected information is shown to the user |
| **Test Status** | | | **Comments** |

| | |
|---|---|
| 1. Pass<br>2. Pass<br>3. Pass<br>4. Pass<br>5. Pass | |

| **Test Case ID** | UAT014 | **Test Name** | Test delete specific existing diet plan |
|---|---|---|---|
| **Test Case Description** | | Examine whether the system able to delete the chosen existing diet plan | |
| **Test Case Scenario** | | | **Test Data** |
| 1. User navigates to view diet plan screen.<br>2. User presses to view the specific diet plan.<br>3. User presses the delete icon button. | | | |
| **Expected Result** | | | **Actual Result** |
| The selected diet plan is deleted from the firestore. | | | The selected diet plan is deleted from the firestore. |
| **Test Status** | | | **Comments** |
| 1. Pass<br>2. Pass<br>3. Pass<br>4. Pass<br>5. Pass | | | |

| **Test Case ID** | UAT015 | **Test Name** | Test generate diet plan |
|---|---|---|---|
| **Test Case Description** | | Examine whether the system able to prompt the diet plan information and update the latest diet plan to the database | |
| **Test Case Scenario** | | | **Test Data** |

| | |
|---|---|
| 1. User navigates to view diet plan screen.<br>2. User chooses the goal preferences.<br>3. User presses on the get diet plan button | Goal Preferences: "Losing Weight" |
| **Expected Result** | **Actual Result** |
| The system will generate the diet plan. | The system will generate the diet plan. |
| **Test Status** | **Comments** |
| 1. Pass<br>2. Pass<br>3. Pass<br>4. Pass<br>5. Pass | |

| | | | |
|---|---|---|---|
| **Test Case ID** | UAT016 | **Test Name** | Test generate diet plan without an existing progress data |
| **Test Case Description** | | Examine whether the system able to show message | |
| **Test Case Scenario** | | | **Test Data** |
| 1. User navigates to view diet plan screen.<br>2. User chooses the goal preferences.<br>3. User presses on the Generate button. | | | Goal Preferences: "Losing Weight" |
| **Expected Result** | | | **Actual Result** |
| "No progress data detected" | | | "No progress data detected" |
| **Test Status** | | | **Comments** |
| 1. Pass<br>2. Pass<br>3. Pass<br>4. Pass<br>5. Pass | | | |

| **Test Case ID** | UAT017 | **Test Name** | Test record feedback |
|---|---|---|---|
| **Test Case Description** | | Examine whether the system able to retrieve user's feedback | |
| **Test Case Scenario** | | | **Test Data** |
| 1. User navigates to feedback screen.<br>2. User fills in the feedback form.<br>3. User presses on the submit button | | | Name: "Shiin Wei"<br>Email: "tnekshiinwei@gmail.com<br>Subject: "Testing"<br>Comment: "Testing ya" |
| **Expected Result** | | | **Actual Result** |
| "Feedback submitted successfully" | | | "Feedback submitted successfully" |
| **Test Status** | | | **Comments** |
| 1. Pass<br>2. Pass<br>3. Pass<br>4. Pass<br>5. Pass | | | |

| **Test Case ID** | UAT018 | **Test Name** | Test record feedback with blank input |
|---|---|---|---|
| **Test Case Description** | | Examine whether the system able to show message | |
| **Test Case Scenario** | | | **Test Data** |
| 1. User navigates to feedback screen<br>2. User presses on the submit button | | | |
| **Expected Result** | | | **Actual Result** |
| "Please enter your name"<br>"Please enter your email"<br>"Please enter your subject"<br>"Please enter your comment" | | | "Please enter your name"<br>"Please enter your email" |

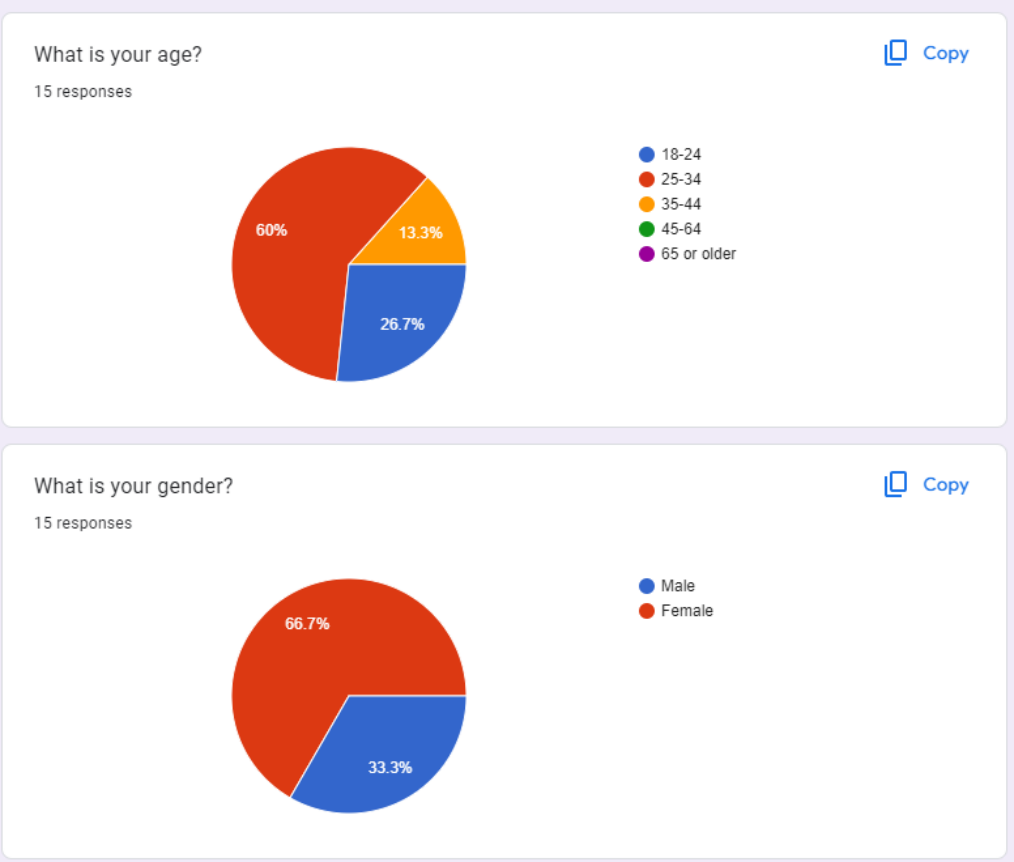
| | |
|---|---|
| | "Please enter your subject"<br>"Please enter your comment" |
| **Test Status** | **Comments** |
| 1. Pass<br>2. Pass<br>3. Pass<br>4. Pass<br>5. Pass | |

| **Test Case ID** | UAT019 | **Test Name** | Test diet chatbot with a valid question |
|---|---|---|---|
| **Test Case Description** | | Examine whether the system able to prompt response according to user input | |
| **Test Case Scenario** | | | **Test Data** |
| 1. User navigates to chatbot screen<br>2. User input their questions and presses on the send button. | | | Message: "Please help me on my diet" |
| **Expected Result** | | | **Actual Result** |
| The chatbot will give response according to the message content. | | | The chatbot will give response according to the message content. |
| **Test Status** | | | **Comments** |
| 1. Pass<br>2. Pass<br>3. Pass<br>4. Pass<br>5. Pass | | | |

| **Test Case ID** | UAT020 | **Test Name** | Test diet chatbot with a blank input |
|---|---|---|---|

| **Test Case Description** | Examine whether the system able to show error message |
|---|---|
| **Test Case Scenario** | **Test Data** |
| 1. User navigates to chatbot screen<br>2. User presses on the send button. | |
| **Expected Result** | **Actual Result** |
| "Please enter a valid question." | "Please enter a valid question." |
| **Test Status** | **Comments** |
| 1. Pass<br>2. Pass<br>3. Pass<br>4. Pass<br>5. Pass | |

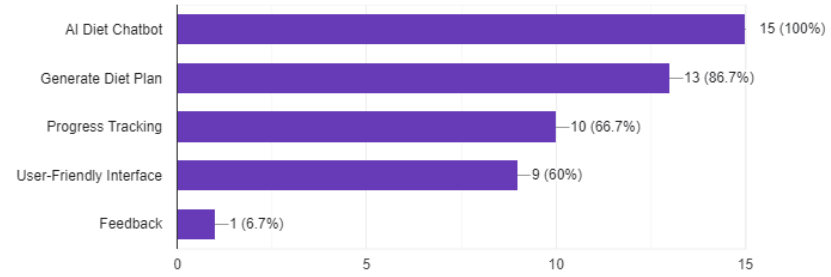
Appendix E: User Acceptance Test Feedback Result



## Satisfaction Survey on DietAI Application

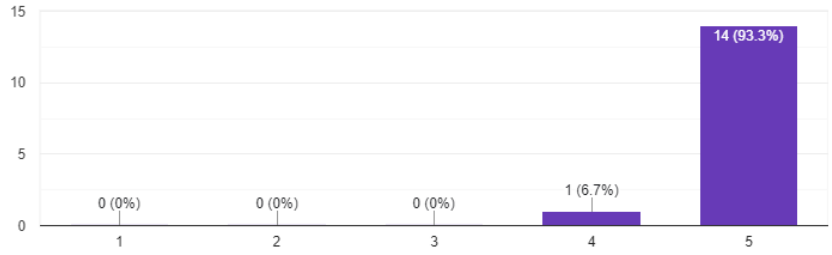
**What features of this application do you find most valuable? (Select all that apply)** Copy

15 responses



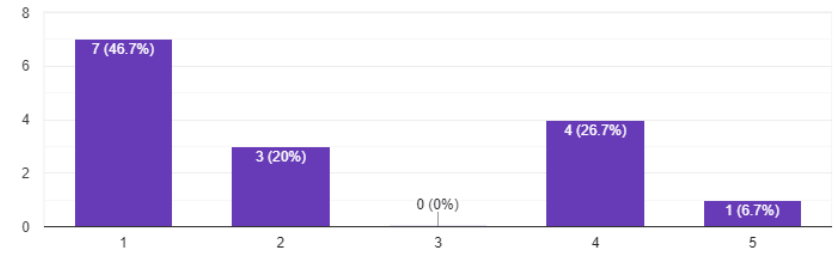

I think that I would like to use this application to keep a good diet. Copy

15 responses



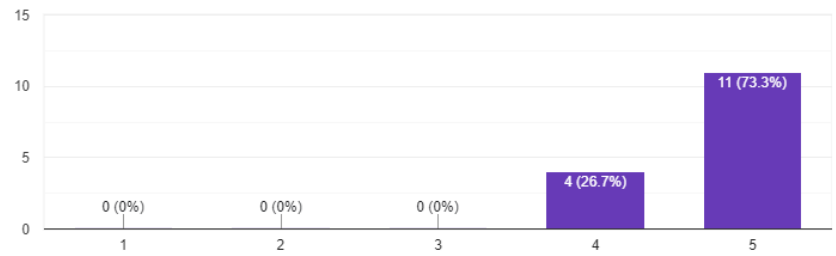

I found the application unnecessarily complicated. Copy

15 responses




The application was easy to use. Copy

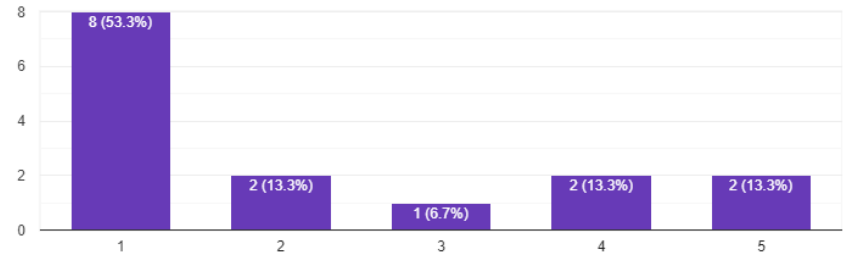
15 responses



I think that there are too many inconsistencies in this application.

Copy

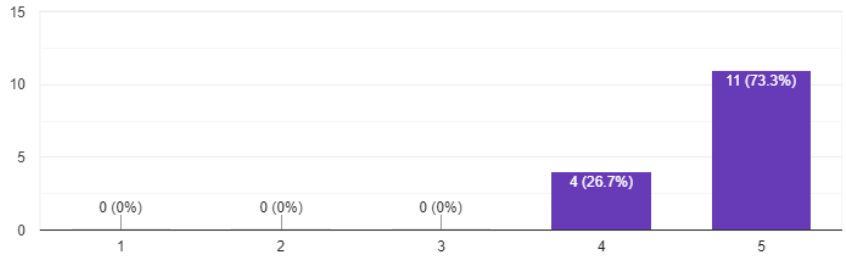
15 responses




**How would you rate your overall satisfaction with this application?**

Copy

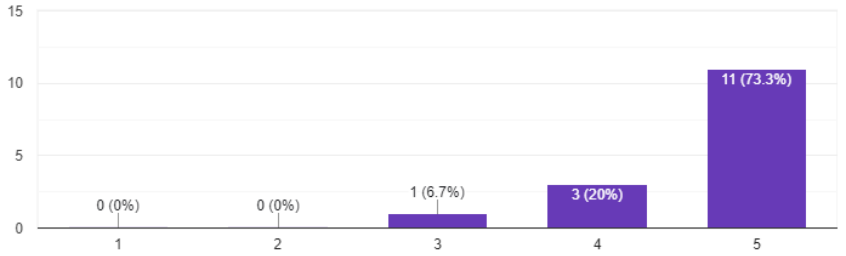
15 responses




**How likely are you to recommend this application to a friend or colleague?**

Copy

15 responses




**I am looking forward for using this application further**

Copy

15 responses