

Personal Nutritionist Application for Diet recommendation system based on user health information

Submitted in partial fulfillment of the
requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering

By

Bedugam Kalyan Kumar (Reg.No - 39110147)

Pallaka Venkata Narendra (Reg.No – 39110732)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade “A” by NAAC
JEPPIAAR NAGAR, RAJIV GANDHISALAI,
CHENNAI – 600119**

APRIL - 2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with —A|| grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Bedugam Kalyan Kumar (Reg.No:-39110147)** and **Pallaka Venkata Narendra (Reg.No:-39110732)** who carried out the Project Phase-2 entitled “**Personal nutritionist Application for Diet recommendation system based on user health information**” under my supervision from January 2023 to April 2023.

Internal Guide

Dr. JEMSHIA MIRIAM, M.E., Ph.D.



Head of the Department

Dr. L. LAKSHMANAN, M.E., Ph.D.

Submitted for Viva voce Examination held on 24.04.23

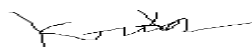
Internal Examiner

External Examiner

DECLARATION

I **Bedugam Kalyan Kumar**(Reg.No -39110147) hereby declare that the Project Phase-1 Report entitled “**Personal nutritionist Application for Diet recommendation system based on user health information**” done by me under the guidance of **Dr. jemshia Miriam M.E.,Ph.D.**, is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in **Computer Science and Engineering**.

DATE: 24.04.23



PLACE:Chennai

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D., Dean**, School of Computing, **Dr. L. Lakshmanan M.E., Ph.D.**, Head of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr JEMSHIA MIRIAM., Ph.D.**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my phase-1 project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

In today's lifestyle, people are moving towards achieving a fit and healthy body. This shift has changed the way people are living right now in almost every household. Hence, healthy eating and nutritious food have become an essential part of everyone's lifestyle to achieve a balanced and healthy life in such busy and hectic environment. On the other hand, an imbalanced diet can lead to disastrous results on one's health, which may lead to diseases such as obesity, diabetes, etc. However, such conventional diseases and their symptoms can be reduced or prevented by active living and by including better nutritional diet. Hence, those in search of healthy and nutritious food have used the internet to research the food's nutrition values. A Health balanced diet is important because organs and tissues need proper nutrition to work effectively. Calories play a vital role in our growth and energy. A good diet can help you manipulate calorie intake based on your requirements. This application provides the user with a complex algorithm which can provide the user with a Health diet plan based on his/her characteristics like height, weight, BMI, gender etc. If the user is allergic to some kind of food, it also has the feature to contact an actual dietitian to consult. And there's also a page where the user can just read some interesting facts on health and human body.

TABLE OF CONTENT

Chapter No	TITLE	Page No.
	ABSTRACT	v
	LIST OF FIGURES	1
1	INTRODUCTION	2
	1.1 GENERAL	2
	1.2 BALANCED DIET	4
2	LITERATURE SURVEY	12
	2.1 INTERFACES FROM LITERATURE SURVEY	15
	2.2 OPEN PROBLEMS IN EXISTING SYSTEM	15
3	SYSTEM ANALYSIS	17
	3.1 EXISTING SYSTEM	17
	3.2 PROPOSED SYSTEM	17
	3.3 FEASIBILITY STUDY	18
	3.4 REQUIREMENT SPECIFICATION	19
	3.5 LANGUAGE SPECIFICATION	20
4	Description of Proposed System	24
	4.1 SELECTED METHODOLOGY OR PROCESSMODEL	24
	4.2 ARCHITECTURE / OVERALL DESIGN OF PROPOSED SYSTEM	26
	4.3 DESCRIPTION OF SOTWARE FOR IMPLEMENTATION	28
	4.4 FINACIAL REPORT ON ESTIMATED COSTING	30
	4.5 TRANSITION/SOFTWARE TO OPERATIONS PLAN	31
5	IMPLEMENTATION DETAILS	33
	5.1 DEVELOPMENT AND DEPLOYMENT SETUP	33

	5.1.1	DEVELOPMENT SETUP	33
	5.1.2	DEPLOYMENT SETUP	33
	5.2	ALGORITHMS	34
	5.3	TESTING	36
6		RESULTS AND DISCUSSION	38
7		CONCLUSION	41
	7.1	CONCLUSION	41
	7.2	FUTURE WORK	43
	7.3	RESEARCH ISSUES	43
	7.4	IMPLEMENTATION ISSUES	45
		REFERENCES	47
		APPENDIX	49
	A.	SOURCE CODE	49
	B.	SCREENSHOTS	80
	C.	RESEARCH PAPER	82

LIST OF FIGURES

FIGURE NO	FIGURE NAME	Page. No.
1.1	Healthy Food Pyramid	5
1.2	Flutter System Overview	8
1.3	Flutter Architecture	9
4.1	Katch-McArdle Multipliers for Calculating TDEE	26
4.2	System Architecture	26
4.3	Block diagram of Application	27
6.1	Screenshot of Diet-page	40
8.1	Screenshot of login page and sign up page	79
8.2	Screenshot of input details of user	80
8.3	Screenshot of Homepage of application	80

CHAPTER 1

INTRODUCTION

1.1 GENERAL

The main purpose of this Project is to get healthy diet for people who are difficulty in visiting for their regular health checkups and ask for their regular nutritional balance. They always need an efficient source that is time saving, costless and provide them an easy way for nutritional assessment on their doorsteps to maintain a balance nutrition and healthy life, therefore, keeping in view the above, the use of smart gadget and mobile apps and their progression in innovation plays a strong role in maintaining a balance nutrition and to have enough knowledge about the usage and function of collected nutritional apps on a single click. This application includes a variety of nutritional diet along with diet coach, food diary, weight loss, calories counter and pregnancy diet. Along with this, it is mandatory to know the basic knowledge of human nutrition. Human Nutrition the area of Medical Sciences which deals with nutrients and their influences on human routine eating, health and exercises. Nutrition or nutrients provide growth and development to human beings as well as to plants. In Nutrition mostly appropriate diet is that which contains the micro and macronutrients. According to the Flurry Analytics, “the mobile development world saw a sudden 62% increase in the usage of health apps in 2016”. In the Android market, there are many healthcare applications available that provide diet plans for their clients. However, there is no such app available that can provide diet plan as well as activity tracker functionality to their user. Hence, to make their fitness path a bit smoother and to enhance their experience, we have built an Android application, Personal Dietitian, to provide a broader approach in providing a healthier living through nutritious diet plan to the users. Basal Metabolic Rate (BMR) is the number of calories you burn as your body performs basic (basal) life-sustaining function The Harris-Benedict Equation was one of the earliest equations used to calculate basal metabolic rate (BMR)

MEM:BMR = 88.362 + (13.397 x weight in kg) + (4.799 x height in cm) – (5.677 x age in years)

Women: BMR = 447.593 + (9.247 x weight in kg) + (3.098 x height in cm) – (4.330 x age in years)

Once the calories are calculated then the system decides if the user is in a under-weight, healthy or overweight category based on his BMI. Based on the category selected the diet is provided by the application to the user. The Diet is based on the charts of diets calculated & created by expert Dietitians and Nutritionist. We give user to register with the Personal Dietitian application .Special diet for abnormal Menstruation women, pregnant women, Diabetic patients. Count burnt and eaten calories (Integration of Wearable Device).

Inadequate and inappropriate intake of food is known to cause various health issues and diseases. Due to lack of concise information about healthy diet, people have to rely on medicines instead of taking preventive measures in food intake. Due to diversity in food components and large number of dietary sources, it is challenging to perform real-time selection of diet patterns that must fulfill one's nutrition needs. Particularly, selection of proper diet is critical for patients suffering from various diseases. Recommender system has been widely used in recent days, in the field of food recommendation. Much of the attention in the diet and nutrition is being paid to diet management systems, which have been replacing traditional paper-and-pen methods. These systems include informative content and services, which persuade users to alter their behavior. Due to the popularity of these diet monitoring facilities, these systems hold a vast amount of user preference information, which could be harnessed to personalize interactive features and to increase engagement with the system and the diet program. One such personalized service, ideally suited to informing diet, is a food recommender. This recommender could exploit the

nutritional values of the food to inform its recommendations. The goal of the application is to provide a platform where users find their favorite food and its nutritional value. This is useful for anyone who is health conscious or wants to lose weight. This application can be used as a standalone application or it can also be used as a part of a more sophisticated application. The application can help people

who like to eat milk or fish. The application is targeted towards a local audience, for now, and as of present it can only be used as a web application that recommends food based in their nutritional value

1.2 Balanced Diet

A healthy balanced diet meets each of a person's nutrient needs. To remain healthy, humans require a specific number of calories and nutrients. A balanced diet provides an individual all the nutrition they need without exceeding the daily calorie allowance. People can consume the nutrients and calories they require by eating a balanced diet while avoiding junk food or other foods with low nutritional value.

Previously, the United States Department of Agriculture (USDA) advised adhering to the food guide pyramid. Yet, given the advancements in nutritional research, it is now advised to construct a healthy plate using items from all five food categories. The USDA recommends that veggies and fruits should comprise half of a human's plate. Grain and protein should make up the remaining half. They advise including a dish of low-fat dairy or another source of the nutrients present in dairy with each meal

Elements Of a Balanced and Nutritious Diet



Fig 1.1 Healthy Food Pyramid

A nutritious diet should include items from all five of these food groups:

- **Vegetables**

People should select a variety of veggies to receive essential nutrients and prevent dietary monotony. Furthermore, the USDA advises Trusted Source persons to consume veggies from each of the five groupings at least once each week. Vegetables can be eaten either raw or cooked.

It's crucial to keep in mind nevertheless that preparing veggies depletes them somewhat nutritionally. Additionally, some techniques, like deep-frying, can add unhealthy fats to a dish. The vegetable group includes five subgroups: leafy greens, starchy vegetables, red or orange vegetables, other vegetables such as eggplant or zucchini, and beans and peas.

- **Fruits**

Fruits are wholesome, create a delectable snack or dessert, and can quell a sweet desire. Seasonal fruits grown locally are more nutrient-dense and fresher than foreign fruits. Fruits have a lot of sugar, but it's natural sugar. Fruits also include fiber and other micronutrients, unlike candy and many sugary pastries.

This indicates that they will increase the bodies natural supply of vital vitamins, minerals, and antioxidants while being less probable to occur in a sugar rise. Your doctor or nutritionist can provide guidance on the best fruits to choose, how much to consume, and when if you have diabetes.

- **Grains**

Although refined white flour is used in many breads and baked items, it is not particularly nutritious. This is due to the fact that a large portion of the nutrition is found in the grain's hull, or outer shell, which manufacturers remove during processing. Products made from whole grains contain the complete grain, including the hull. They add extra fibre, vitamins, and minerals.

Whole grains are frequently thought to enhance the flavour and texture of a dish. A helpful diet suggestion is to try moving from white breads, pastas, and rice to whole grain alternatives.

- **Protein**

According to the 2015-2020 Dietary Guidelines for Americans, everyone should regularly consume protein that is high in nutrients. According to the recommendations, a human's dish should contain a fifth of this macronutrient.

Lean beef and pork, chicken and turkey, fish, beans, peas, and legumes are all good sources of protein. Protein is mostly found in meat and soybeans and is necessary for a number of processes, including muscle growth and development.

- **Dairy**

Products made from dairy and fortified soy are essential sources of calcium. Whenever practical, the USDA advises choosing low-fat varieties.

Ricotta or cottage cheese, low-fat milk, yoghurt, and soy milk are examples of low-fat cheese and soy goods. People who are lactose intolerant can pick low-lactose or lactose-free products or calcium as well as other dietary components based on soy.

1.2.1 Diet Recommendation System

The term "nutrition informatics" refers to the intersection of information technology (IT), information sciences, and nutrition. Registered dietitians and dietetic technicians can now practice in this area and generate income for health care using this innovative area of health informatics. The phrase "nutrition informatics" was first used in a scientific context in 1996, although nutritionists have been using technologies and implementing nutritional values for decades.

In 1962, the first publication demonstrating the possible application of computers to analyze dietary consumption appeared in print. IT has been used in the healthcare field to promote medical studies through the recycling of data, establish patient care through the use of electronic health records, and gather population figures through gathering of personal health information.

Additionally, specific electronic technologies have been used by dietitians to manage patient tray service, indexing, and nutrient assessment. Despite the fact that most people are aware of the importance of maintaining nutritious eating habits, urban lifestyles and/or a lack of motivation to invest mental effort in provision of food cause them to be more likely to overlook appropriate practices.

People are unable to consume healthful foods because of these obstacles. One of the key technologies used in the field of nutrition informatics is called nutrition recommendations systems (NRS). These are investigated as a useful tool to assist users in altering their eating habits and achieving the goal of making healthier food decisions.

NRS not only suggests users' dietary preferences, but it also suggests alternatives for a healthy diet; in addition, it can suggest a suitable diet and encourage eating behavior, identify health issues, and result in changing user behavior.

In general, recommendation systems have evolved a technology that is efficient and effective for extracting useful information and then using it effectively. A recommender system can suggest new things to consumers and forecast their preferences for unrated goods. Technical specifications and appropriate design based on system kinds and functions determine these systems' capacity. In order to create personalized recommendations, a number of different techniques have been proposed.

These variations in applied techniques and design may result in the creation of different types of recommendation systems, such as collaborative filtering recommender systems (CF), content-based recommender systems (CB), knowledge-based recommender systems (KBS), and hybrid recommender systems (HRS).

There hasn't been a thorough examination into NRS and their traits. Reviewing nutrition recommender system with an emphasis on their traits, varieties, and evaluation techniques is the goal of this work.

1.2.2 Objective of the Project

The project makes use of a dataset that accurately represents the amounts of different nutrients. In response to the circumstance, we have worked to create a

program that advises people to follow a healthy diet. Only three categories—weight loss, weight gain, and healthy—are included in the suggested goods.

In most modern nations, obesity and inactivity are developing issues that place strain on both the public health care system and individual residents. Future efforts to address this issue will benefit from resources that encourage and support individual weight management. But for them to work, they must be accepted by the users and live up to their expectations.

1.3 FLUTTER:

Flutter is an open-source UI software development kit created by Google. It is used to develop cross platform applications for Android, IOS, Linux, macOS, Windows, Google Fuchsia, and the web from a single codebase.

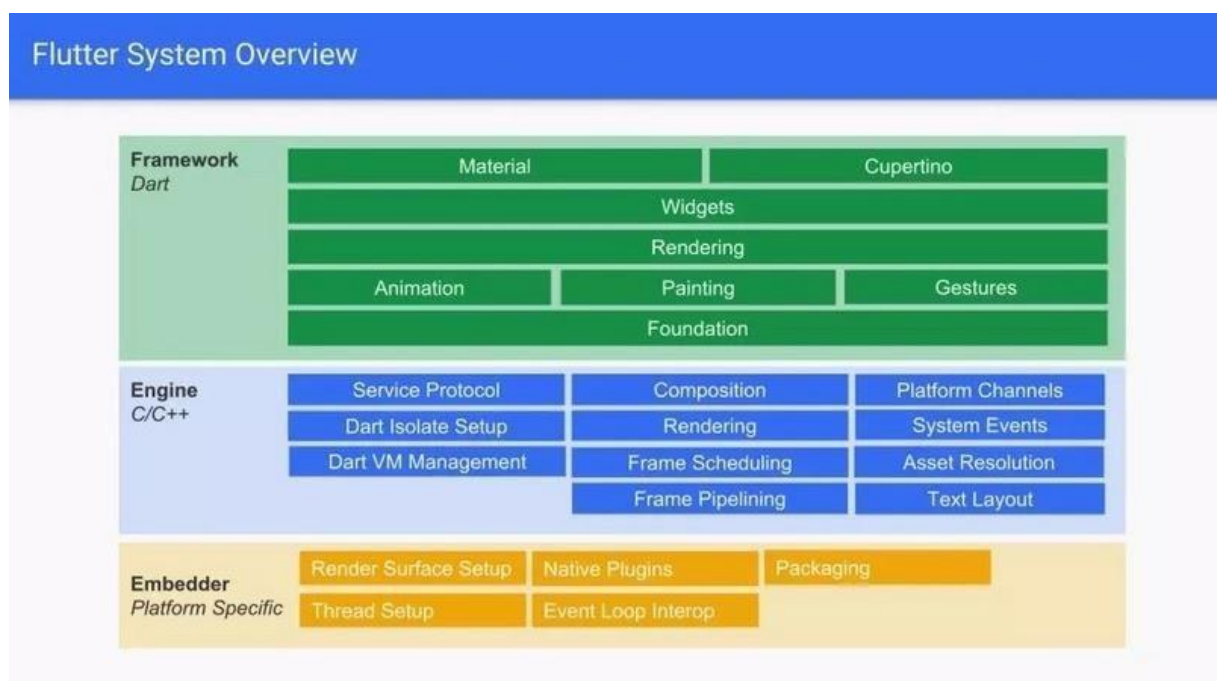


Fig 1.2 Flutter System Overview

Framework architecture

The major components of Flutter include:

- Dart platform
- Flutter engine

- Foundation library
- Design-specific widgets
- Flutter Development Tools (DevTools)

1.4 INTRODUCTION TO DART

Dart is an object-oriented programming language that was first unveiled by Google in 2011

- It is worth mentioning the “dart2native” feature that allows compiling its for Windows, Linux. and macOS platforms as desktop application.
- Dart program can be composed into a self-contained executable file or complied to JavaScript.
- Dart language is easy to study.
- Its syntax is pretty similar to Java, Swift, or Kotlin languages.
- Dart software development kit (SDK) is shipped with a stand-alone Dart Virtual Machine (VM) that allows you to build the code in a command-line interface (CLI) environment

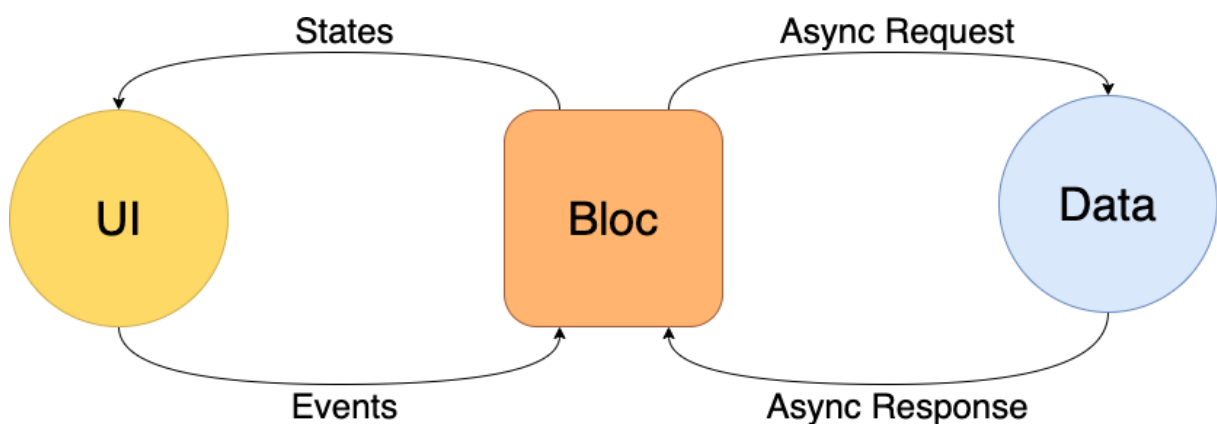


Fig 1.3 Flutter Architecture

CHAPTER 2

LITERATURE SURVEY

[1] H. Jiang, E-C, Chen .Y.-W. Lo, “The application of topic algorithm to diabetic diet recommendation”, January 2015. In application Content-Based Filtering and Collaborative Filtering methods are used to get users choice of his food recommendation for the daily nutrition with help of USDA dataset and grocery data, the nutrition diet recommendations will help the user to maintain and improve their health conditions.

[2] Vijay Jaiswal, “A new approach for recommending healthy diet using predictive data mining algorithm”, IJRAR March 2019 . The studies and their implementation showed that the decision tree learning algorithm, Random Tree works well on any classification problems having a dataset with the non-repeated values. The proper decision regarding which food item should be assigned to a particular individual is done by considering factors such as their likeness for a particular food item, their fitness goals, the content of Nutrients, etc. After that by taking a decision about particular food based on proposed Mathematical Constraints and predefined rules, it is easy to map daily menu planning for an individual.

[3] Xiaoyan Gao, Fuli Feng, Xiangnaa He, Heyan Huang , “Hierarchical Attention Network for Visually-aware Food Recommendation”, Vol. 18 6 Jan 2019 . We proposed a Hierarchical Attention based Food Recommendation (HAFR) system to infer users’ preference over recipes for food recommendation. The HAFR aims to learn more comprehensive recipe representation via jointly leveraging user-recipe interaction history, food image, and food ingredients with a hierarchical attention. We collected a large-scale dataset for food recommendation and conducted extensive experiments, demonstrating that HAFR consistently outperforms the state-of-the-art models.

[4] Raza Yunus, Omar Arif, Hammad Afzal, Muhammad Faisal Amjad, Haider Abbas, "A Framework to Estimate the Nutritional Value of Food in Real Time Using Deep Learning Techniques", IEEE Access, January 2019. Presents a system that exploits the extensive use of mobile devices to provide health information about the food we eat. The mobile based app takes the images of the meal and presents approximate ingredients and nutritional values in food. A dataset is created that consists of common and subcontinental food items. We employ a fine-tuned Inception model to recognize food items and propose methods to estimate attributes of the recognized food item. The results are improved via data augmentation, multi crop evaluation, regularization and other similar techniques.

[5] Bura, D., Choudhary, A., & Singh, R. K. (2017). "A Novel UML Based Approach for Early Detection of Change Prone Classes. International Journal of Open-Source Software and Processes" (IJOSSP), 8(3), 1-23. This article proposes an efficient novel-based approach for predicting changes early in the object-oriented software. Earlier researchers have calculated change prone classes using static characteristics such as source line of code e.g., added, deleted and modified. This research work proposes to use dynamic metrics such as execution duration, run time information, regularity, class dependency and popularity for predicting change prone classes. Execution duration and run time information are evaluated directly from the software. Class dependency is obtained from UML2.0 class and sequence diagrams. Regularity and popularity is acquired from frequent item set mining algorithms and an ABC algorithm. For classifying the class as change-prone or non-change-prone class an Interactive Dichotomize version 3 ID3 algorithm is used.

[6] Godara, D., & Singh, R. K "A review of studies on change proneness prediction in object-oriented software. 2014 international Journal of Computer Applications", 105(3). Systematic review of papers published in journals and conference proceedings. The review investigates methodologies for predicting change prone class and fault prone class. The key findings of the review are: (1) behavior dependency has been widely used for prediction of the change prone class, (2) there is need to develop a framework comprising of more features to accurately

predict change prone class. This paper provides an extensive review of studies related to change proneness of software. The main goal and contribution of the review is to support the research on prediction of change prone classes.

[7] Mohd Afisi, Mohd Shukran, Yuk Ying Chung, "Artificial Bee Colony based Data Mining Algorithms for Classification Tasks", August 2011.

We can provide nutrition recommendations based on BMI calculations in an informative and user-friendly environment. In this food recommendation application, we focus in daily diet plan and nutrition need. According to user food preferences and consumption we get suggestion.

2.1 INFERENCES FROM LITREATURE SURVEY

From the above-mentioned literature works, it is clear that there has been effective research on personal nutritionist and diet recommendation system and many models have been proposed. One of the emergent concerns of human life is about health and wellness. Undeniably, health and nutrition are one of the valuable aspects of life. Thus, technological innovations to help enhance and even promote health awareness is essential.

Mobile applications have the ability to support health needs like detecting heart rate, classifying food, and many more. Taking advantage of technology, utilization of it hereby addresses certain issue and problems of human life, especially in health.

The researcher's attempts to design and develop an Android-based food Diet application that could be used as health awareness tool for non-health-conscious individual. Implementing Mifflin-St Jeor method in determining daily calorie consumption, users shall be aware of their required calorie intake.

2.2 OPEN PROBLEMS IN EXISTING SYSTEM

The existing system is based on content based recommendation systems. Content based food recommender system is where recommend food recipes according to the preferences already given by the user. The preferred recipes of the user are fragmented into ingredients which are assigned ratings according to the stored users' preferences. The recipes with the matching ingredient are recommended. The traditional models do not consider the nutrition factors and the balance in the diet.

Nutrition factors are ignored which are very much important to recommend food and balance diet. Only foods containing milk or fish can be searched. Only displays the nutritional value of the food. Does not contain a wide variety of food but only the popular ones. For the existing system of personalized diet recommendation system, a small dataset is taken and only one or two features such as weight loss taken into consideration. There are many healthcare applications available that provide diet plans for their clients. However, there is no such app available that can provide diet plan as well as activity tracker functionality to their user. Hence, to make their fitness path a bit smoother and to enhance their experience. Where apps like MyFitnessPal provides only calorie count and Healthy Diet app give meal planner, lose weight with die.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

The existing system is based on content based recommendation systems. Content based food recommender system is where recommend food recipes according to the preferences already given by the user. The preferred recipes of the user are fragmented into ingredients which are assigned ratings according to the stored users' preferences. The recipes with the matching ingredient are recommended. The traditional models do not consider the nutrition factors and the balance in the diet. Nutrition factors are ignored which are very much important to recommend food and balance diet. Only foods containing milk or fish can be searched. Only displays the nutritional value of the food. Does not contain a wide variety of food but only the popular ones. For the existing system of personalized diet recommendation system, a small dataset is taken and only one or two features such as weight loss is taken into consideration. The system shows low accuracy and high processing time.

3.2 PROPOSED SYSTEM

This project aims to present the development of an expert system prototype on nutrition and diet domain. The objective of this developed nutrition and diet expert system is to help people to evaluate their nutrition condition and to know their neediness of the type of food and required time to do exercising each day. Moreover, the system provides advices about healthy food and the rate of protein, vitamins, and calcium they have to eat. Accordingly the developed system improves people awareness about the importance of nutrition, reduces consultation time and makes people care more about their health. This developed prototype nutrition expert system provides advice only for healthy people , not for unhealthy people and pregnant and lactating women. The major advantage of the proposed system for personalized nutritionist based system is that it is highly accurate, and precise. For the proposed system, large vast dataset is taken. Features like weight gain are added in proposed system with improved accuracy. The system can be used in

situations where large amounts of data have to be processed in a short amount of time. It is cost-efficient.

3.3 FEASIBILITY STUDY

With an eye towards gauging the project's viability and improving server performance, a business proposal defining the project's primary goals and offering some preliminary cost estimates is offered here. Your proposed system's viability may be assessed once a comprehensive study has been performed. It is essential to have a thorough understanding of the core requirements of the system at hand before beginning the feasibility study. The feasibility research includes mostly three lines of thought:

- Economic feasibility
- Technical feasibility
- Operational feasibility
- Social feasibility

3.3.1 *ECONOMICAL FEASIBILITY*

The study's findings might help upper management estimate the potential cost savings from using this technology. The corporation can only devote so much resources to developing and analysing the system before running out of money. Every dollar spent must have a valid reason. As the bulk of the used technologies are open-source and free, the cost of the updated infrastructure came in far cheaper than anticipated. It was really crucial to only buy customizable products.

3.3.2 *TECHNICAL FEASIBILITY*

This research aims to establish the system's technical feasibility to ensure its smooth development. Adding additional systems shouldn't put too much pressure on the IT staff. Hence, the buyer will experience unnecessary anxiety. Due to the low likelihood of any adjustments being necessary during installation, it is critical that the system be as simple as possible in its design.

3.3.3 OPERATIONAL FEASIBILITY

An important aspect of our research is hearing from people who have actually used this technology. The procedure includes instructing the user on how to make optimal use of the resource at hand. The user shouldn't feel threatened by the system, but should instead see it as a necessary evil. Training and orienting new users has a direct impact on how quickly they adopt a system. Users need to have greater faith in the system before they can submit constructive feedback.

3.3.4 SOCIAL FEASIBILITY

During the social feasibility analysis, we look at how the project could change the community. This is done to gauge the level of public interest in the endeavour. Because of established cultural norms and institutional frameworks, it's likely that a certain kind of worker will be in low supply or nonexistent.

3.4 REQUIREMENT SPECIFICATION

3.4.1 HARDWARE REQUIREMENTS

Processor	: Pentium Dual Core 2.00GHZ
Hard disk	: 120 GB
RAM	: 2GB (minimum)
Keyboard	: 110 keys enhanced

3.4.2 SOFTWARE REQUIREMENTS

Operating system	: Windows7 (with service pack 1), 8, 8.1 and 10
IDE	: Android Studio
Language	: Flutter, Dart

3.5 LANGUAGE SPECIFICATION

Android Studio is the official integrated development environment (IDE) for Android application development. It is based on IntelliJ IDEA, a Java integrated development environment for software, and incorporates its code editing and developer tools.

To support application development within the Android operating system, Android Studio uses a Gradle-based build system, Android Emulator, code templates and GitHub integration. Every project in Android Studio has one or more modalities with source code and resource files. These modalities include Android app modules, Library modules and Google App Engine modules.

Android Studio uses an Apply Changes feature to push code and resource changes to a running application. A code editor assists the developer with writing code and offering code completion, refraction and analysis. Applications built in Android Studio are then compiled into the APK format for submission to the Google Play Store.

The software was first announced at Google I/O in May 2013, and the first stable build was released in December 2014. Android Studio is available for macOS, Windows and Linux desktop platforms. It replaced Eclipse Android Development Tools (ADT) as the primary IDE for Android application development. Android Studio and the Software Development Kit can be downloaded directly from Google.

Flutter is a popular framework for mobile app development that allows developers to create high-performance, cross-platform apps using a single codebase. Android Studio is a popular IDE (Integrated Development Environment) that can be used for Flutter development, providing developers with a powerful set of tools for building, testing, and deploying apps. To use Android Studio for Flutter development, the first step is to install the latest version of Android Studio from the official website. Once the installation is complete, you will need to download and install the Flutter SDK from the official website. After installing the Flutter SDK, make sure to add the Flutter SDK path to your system's environment variables.

Once you have installed Android Studio and the Flutter SDK, you can create a new Flutter project by opening Android Studio and selecting "Start a new Flutter project" from the welcome screen. This will launch the New Flutter Project wizard, which will guide you through the process of setting up your project. You can select a project name, choose your target platform (Android or iOS), and configure your project settings.

After setting up your project, you can start writing code using Android Studio's powerful code editor. Android Studio provides features such as code completion, code formatting, and syntax highlighting, which can help you write high-quality code quickly and easily. You can also use Android Studio's built-in debugging tools to test your app and fix any issues that arise.

Overall, using Android Studio for Flutter development can help you build high-quality, cross-platform apps quickly and easily. By following these steps and leveraging Android Studio's powerful set of tools, you can streamline your Flutter development workflow and bring your app ideas to life.

- **Flutter Inspector:** Android Studio includes a Flutter Inspector tool that allows you to inspect and modify the layout and state of your Flutter widgets in real-time. This can be especially useful for debugging layout issues or understanding how your app is responding to user interactions.
- **Code generation:** Android Studio includes a range of code generation tools that can help you write Flutter code more quickly and easily. For example, you can use the "New Widget" wizard to generate boilerplate code for a new widget, or use the "Extract Method" refactoring to extract common code into a reusable function.
- **Flutter Inspector:** Android Studio includes a Flutter Inspector tool that allows you to inspect and modify the layout and state of your Flutter widgets in real-time. This can be especially useful for debugging layout issues or understanding how your app is responding to user interactions.
- **Code generation:** Android Studio includes a range of code generation tools that can help you write Flutter code more quickly and easily. For example, you can use the "New Widget" wizard to generate boilerplate code for a new widget, or use the "Extract Method" refactoring to extract common code into a reusable function.
- **Integration with other tools:** Android Studio integrates with a range of other tools that can be useful for Flutter development. For example, you can use the Firebase Assistant to add Firebase services to your app, or use the Android Virtual Device Manager to create and manage virtual devices for testing your app.

3.5.1 Activating an Environment

Open Android Studio and navigate to the Flutter SDK installation directory. The default installation location on Windows is C:\src\flutter, while on macOS and Linux it is ~/flutter.

In the Flutter SDK installation directory, navigate to the bin subdirectory.

Open a terminal or command prompt window in this directory. To do this in Android Studio, go to Tools > Terminal > New Terminal.

In the terminal window, run the following command to create a new environment:

```
flutter create --org <your organization name> -a <your project name>
```

This command will create a new Flutter project in the current directory with your specified organization name and project name.

Navigate to the new project directory using the cd command, and then run the following command to activate the environment:

```
"flutter pub get"
```

This command will download and install all the dependencies required by your project.

You can now start developing your Flutter app in the newly created virtual environment. To activate the environment for an existing project, you can use the same flutter pub get command in the project directory to download and install the dependencies.

3.5.2 Installing Packages in an Environment

Open a terminal or command prompt window in this directory. To do this in Android Studio, go to Tools > Terminal > New Terminal.

In the terminal window, run the following command to activate the environment:

```
'flutter pub get'
```

This command will download and install all the packages that are listed in your project's pubspec.yaml file. The packages are installed in the pub-cache subdirectory of your project directory. To install a new package, update the dependencies section

of your pubspec.yaml file with the name of the package and its version number. For example:

Dependencies:

Flutter

sdk: flutter

cupertino_icons: ^1.0.2

In this example, the cupertino_icons package is added with version number ^1.0.2. Save the pubspec.yaml file and run the flutter pub get command again in the terminal window. This will download and install the new package and any of its dependencies. You can now import the package in your Flutter code and use its functionality. For example:

```
import 'package:flutter/material.dart';
```

```
import 'package:cupertino_icons/cupertino_icons.dart';
```

In this example, the cupertino_icons package is imported and used to add the CupertinoIcons.back icon to the app's AppBar.

3.5.3 Updating Packages in an Environment

In the terminal window, run the following command to activate the environment:

```
flutter pub upgrade
```

This command will update all the packages that are listed in your project's pubspec.yaml file to the latest compatible versions. The updated packages are installed in the pub-cache subdirectory of your project directory. If you want to update a specific package to a specific version, you can specify it in the pubspec.yaml file. For example:

dependencies:

flutter:

```
sdk: flutter
```

```
cupertino_icons: ^1.0.3
```

In this example, the `cupertino_icons` package is updated to version 1.0.3. Save the `pubspec.yaml` file and run the `flutter pub upgrade` command again in the terminal window. This will download and install the updated package and any of its dependencies. You can now import the updated package in your Flutter code and use its updated functionality.

3.5.4 Exporting an Environment Configuration

Open a terminal or command prompt window in this directory. To do this in Android Studio, go to `Tools > Terminal > New Terminal`.

In the terminal window, run the following command to activate the environment:

```
flutter packages get
```

This command will ensure that all the packages that are listed in your project's `pubspec.yaml` file are installed.

Run the following command to export the environment configuration to a file:

```
flutter packages pub run environment_config:export
```

This command will generate a file named `environment_config.yaml` in your project directory. This file contains the environment configuration for your project.

Share the `environment_config.yaml` file with others who need to replicate your development environment. To import the environment configuration from the `environment_config.yaml` file, the other person can run the following command in their terminal window: `flutter packages pub run environment_config:import`. This command will import the environment configuration from the `environment_config.yaml` file and install all the necessary packages.

3.5.5 Removing a Package from an Environment

Open a terminal or command prompt window in this directory. To do this in Android Studio, go to Tools > Terminal > New Terminal.

In the terminal window, run the following command to activate the environment:

```
flutter packages get
```

This command will ensure that all the packages that are listed in your project's pubspec.yaml file are installed.

Open the pubspec.yaml file and remove the package that you want to delete from the dependencies section.

For example, if you want to remove the cupertino_icons package, your pubspec.yaml file might look like this:

dependencies:

flutter:

sdk: flutter

Save the pubspec.yaml file and run the following command in the terminal window to remove the package:

```
flutter packages pub remove cupertino_icons
```

This command will remove the cupertino_icons package and any of its dependencies.

You can now verify that the package has been removed by running the following command:

```
flutter packages get
```

This command will ensure that all the packages that are listed in your project's pubspec.yaml file are installed, and will remove any packages that are no longer needed.

3.5.6 Deleting an Environment

Open a terminal or command prompt window in this directory. To do this in Android Studio, go to Tools > Terminal > New Terminal.

In the terminal window, run the following command to list all of the environments that are currently installed:

```
flutter environments
```

This command will display a list of all the environments that are installed on your computer.

Identify the name of the environment that you want to delete from the list.

Run the following command to delete the environment:

```
flutter sdk-path --delete --android
```

Replace `sdk-path` with the path to the environment that you want to delete. For example, if you want to delete an environment located at `/Users/YourUserName/flutter_env`, the command would be:

```
flutter /Users/YourUserName/flutter_env --delete --android
```

Press Enter to execute the command.

The environment will be deleted, and you will no longer be able to use it for your Flutter project.

CHAPTER 4

DESCRIPTION OF PROPOSED SYSTEM

4.1 SELECTED METHODOLOGY OR PROCESS MODEL

4.1.1 Data Collection

The module majorly deals with gathering the users the personal information such as height, weight, food habits etc. which is then stored into a data set. Once that's done the data set is launched and the basic functions / methodologies are derived from it. The Formulas used for calculating the BMI (Body mass Index) and to calculate the total calories to intake was found on internet on some research.

Dataset will be gathered from USDA database who is responsible for storing food nutrition information. Grocery data is created by user as per his food interest. As a first level Quetelet Index (BMI) and food consumed in that day from the grocery data will be taken as input and at second level we will query nutrition information for the user pantry data from the USDA database. According to user's BMI food nutrition's are recommended to complete the nutrition food for user interest and health conditions. The deficit nutrition food, food items, food list, tracking of every day nutrition factors, suggestions, and symptoms are provided in this nutrition diet recommendation system android application. The collaborative filtering method do predictions based on the user's grocery data, USDA API datasets and deficient nutrients. Content-based filtering recommends food items based on a comparison between the content of the food items and a user's pantry data.

$$\text{BMI} = \frac{\text{weight in kg}}{\text{height}^2 \text{ in m}}$$

4.1.2 BASAL METABOLIC RATE(BMR)

Basal Metabolic Rate (BMR) is the number of calories you burn as your body performs basic (basal) life-sustaining function

The Harris-Benedict Equation was one of the earliest equations used to calculate basal metabolic rate (BMR)

Men	$BMR = 88.362 + (13.397 \times \text{weight in kg}) + (4.799 \times \text{height in cm}) - (5.677 \times \text{age in years})$
Women	$BMR = 447.593 + (9.247 \times \text{weight in kg}) + (3.098 \times \text{height in cm}) - (4.330 \times \text{age in years})$

Men	$BMR = 88.362 + (13.397 \times \text{weight in kg}) + (4.799 \times \text{height in cm}) - (5.677 \times \text{age in years})$
Women	$BMR = 447.593 + (9.247 \times \text{weight in kg}) + (3.098 \times \text{height in cm}) - (4.330 \times \text{age in years})$

- W is body weight in kg
- H is body height in cm
- A is age

Once the calories are calculated then the system decides if the user is in a under-weight, healthy or overweight category based on his BMI. Based on the category selected the diet is provided by the application to the user. The Diet is based on the charts of diets calculated & created by expert Dietitians and Nutritionists

TDEE must be calculated to determine the number of calories expend and the amount needed to eat in a daily basis. TDEE is determined by four key factors of expending energy: Basal Metabolic Rate (BMR), Thermic Effect of Food (TEF), Non-Exercise Activity Thermogenesis (NEAT), and Thermic Effect of Activity (TEA). BMR is the energy expenditure to keep the body functioning at rest like using such energy for vital organs. The BMR calculation is different for men (eq. [1]) and women (eq. [2]). TEF, on the other hand, is used for digesting food and nutrients. NEAT is the energy expended for spontaneous activities (e.g., typing, working, and fidgeting). Lastly, TEA is contrasting to NEAT as it is the number of calories burned as a result of exercise. TDEE is the sum of these four factors, see equation (3), that will result to an estimate of the calorie amount needed on a daily basis in order to maintain the current weight. While the BMR commonly utilized Harris-Benedict Equation, the TEF, TEA, and NEAT are combined together and calculated using Katch - McArdle multipliers [32] as shown on Figure 2. In the BMR formulas for men and women, W stands for Weight in kilogram, H is height in centimeter, and A is Age in years.



Fig 4.1 Katch-McArdle Multipliers for Calculating TDEE

$$\text{Men BMR} = 66 + (13.7 \times W) + (5 \times H) - (6.8 \times A) \quad (1)$$

$$\text{Women BMR} = 655 + (9.6 \times W) + (1.8 \times H) - (4.7 \times A) \quad (2)$$

$$\text{TDEE} = \text{BMR} + \text{TEF} + \text{NEAT} + \text{TEA} \quad (3)$$

4.2 ARCHITECTURE / OVERALL DESIGN OF PROPOSED SYSTEM

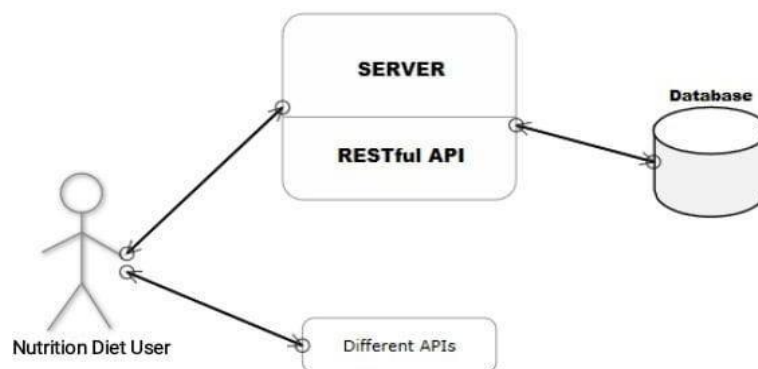


Fig 4.2 System Architecture

The application architecture is based on the client-server architecture. In the NUTRITION DIET Mobile Application, the user acts as a client who is consuming services from the server side of the application, where the user data stored in the Firebase which is NoSql and real-time hosting of databases.

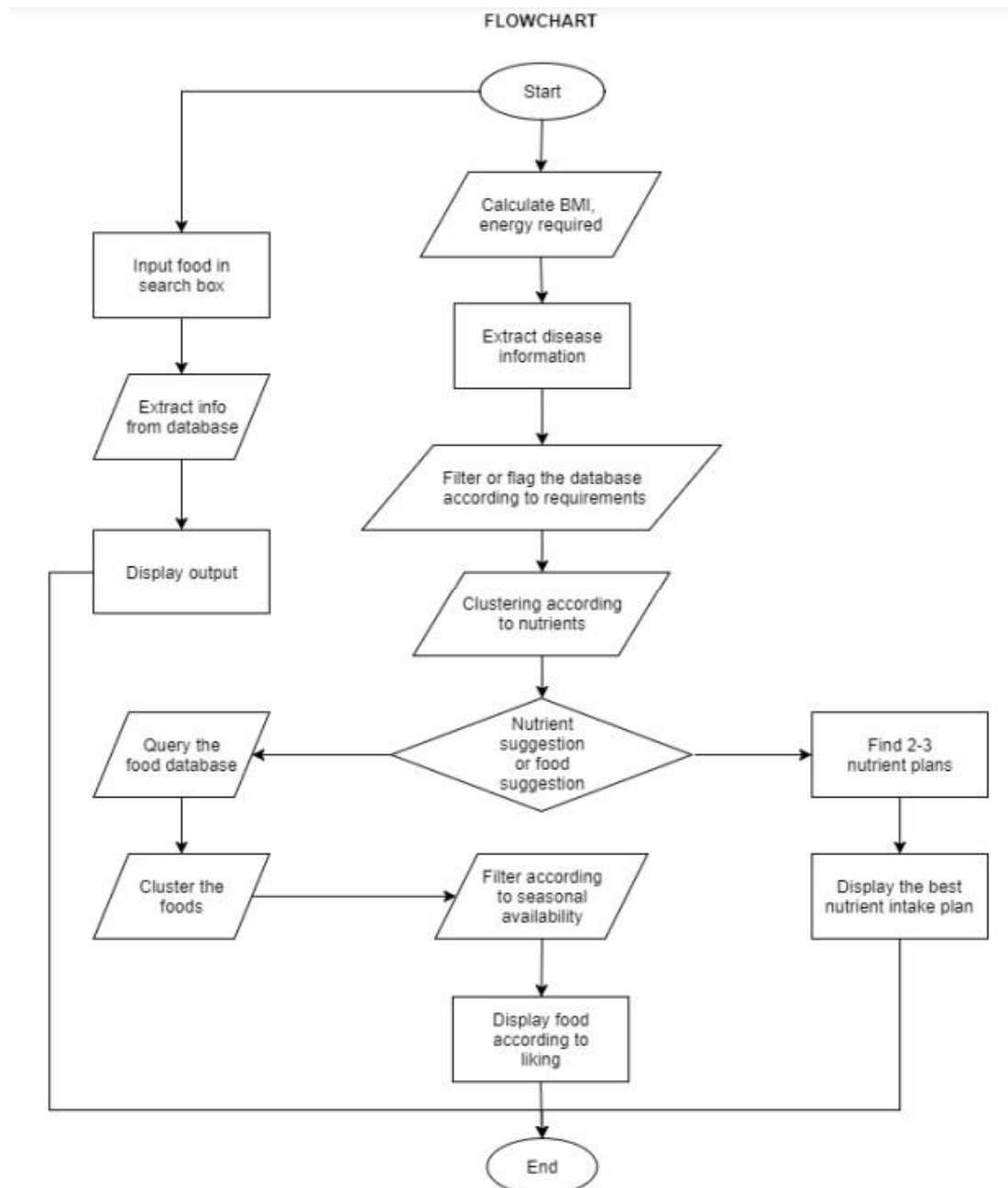


Fig 4.2 Block diagram of Application

4.3 DESCRIPTION OF SOFTWARE FOR IMPLEMENTATION AND TESTING PLAN OF THE PROPOSED MODEL/SYSTEM

4.3.1 *Android Studio*

It is the official integrated development environment (IDE) for Google's for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems. It is a replacement for the Eclipse Android Development Tools (E-ADT) as the primary IDE for native Android application development. Android Studio was announced on May 16, 2013, at the Google I/O conference. It was in early access preview stage starting from version 0.1 in May 2013, then entered beta stage starting from version 0.8 which was released in June 2014. The first stable build was released in December 2014, starting from version 1.0. At the end of 2015, Google dropped support for Eclipse ADT, making Android Studio the only officially supported IDE for Android development. On May 7, 2019, Kotlin replaced Java as Google's preferred language for Android app development. Java is still supported, as is C++.

Android Studio supports all the same programming languages of IntelliJ (and C Lion) e.g. Java, C++, and more with extensions, such as Go; and Android Studio 3.0 or later supports Kotlin and "all Java 7 language features and a subset of Java 8 language features that vary by platform version." External projects backport some Java 9 features. While IntelliJ states that Android Studio supports all released Java versions, and Java 12, it's not clear to what level Android Studio supports Java versions up to Java 12 (the documentation mentions partial Java 8 support). At least some new language features up to Java 12 are usable in Android.

4.3.2 Android Studio Installation

- So let's launch *Android Studio.exe*, Make sure before launch Android Studio, Our Machine should required installed Java JDK. To install Java JDK, take a references of Android environment setup.
- Once you launched Android Studio, its time to mention JDK7 path or later version in android studio installer.
- Need to check the components, which are required to create applications, below the image has selected Android Studio, Android SDK, Android Virtual Machine and performance (Intel chip).
- Need to specify the location of local machine path for Android studio and Android SDK, below the image has taken default location of windows 8.1 x64 bit architecture.
- Need to specify the ram space for Android emulator by default it would take 512MB of local machine RAM.
- At final stage, it would extract SDK packages into our local machine, it would take a while time to finish the task and would take 2626MB of Hard disk space.
- After done all above steps perfectly, you must get finish button and it gonna be open android studio project with Welcome to android studio message as shown below.
- You can start your application development by calling start a new android studio project. in a new installation frame should ask Application name, package information and location of the project.
- After entered application name, it going to be called select the form factors your application runs on, here need to specify Minimum SDK, in our tutorial, I have declared as API23: Android 6.0(Mashmallow).
- The next level of installation should contain selecting the activity to mobile, it specifies the default layout for Applications
- At the final stage it going to be open development tool to write the application code.
-

4.3.3 Create Android Virtual Device

- To test your Android applications, you will need a virtual Android device. So before we start writing our code, let us create an Android virtual device. Launch Android AVD Manager Clicking AVD_Manager icon as shown below.
- After Click on a virtual device icon, it going to be shown by default virtual devices which are present on your SDK, or else need to create a virtual device by clicking

4.3.4 Create new Virtual device button

- If your AVD is created successfully it means your environment is ready for Android application development. If you like, you can close this window using top-right cross button. Better you re-start your machine and once you are done with this last step, you are ready to proceed for your first Android example but before that we will see few more important concepts related to Android Application Development.
- Need to run the program by clicking **Run>Run App** or else need to call **shift+f10** key. Finally, result should be placed at Virtual devices as shown below

How to Use Android Studio for Flutter

Install the Flutter and Dart plugins:

After the successful installation of Android Studio, you have to install Flutter and Dart plugins. To do so follow the steps mentioned below:

- 1 **Start** Android Studio.
- 2 Open plugin preferences (**Configure > Plugins as of v3.6.3.0 or later**).
- 3 Select the **Flutter** plugin and click Install.
- 4 Click Yes when prompted to install the **Dart** plugin.
- 5 Click **Restart** when prompted.

4.4 Financial report on estimated costing

Software and Hardware Costs:

Using open-source software and hardware can significantly reduce the cost of development. Here's a breakdown of the estimated software and hardware costs:

Operating System: Windows(open-source) -Free

Integrated Development Environment (IDE)- Android studio(open-source)-Free

Backend cloud computing services – Firebase (no-cost tier pricing plan) - Free

Development Hardware: Laptop - Assuming you already have a laptop, there will be no additional cost.

Total software and hardware costs: Free (assuming no hardware purchases are needed)

Operational Costs

The operational costs for a Personal nutritionist Application for Diet recommendation system based on user health information for will include ongoing costs such as server maintenance, customer support, and marketing. Assuming you are developing the system on your own and have a small user base, the estimated operational costs for the first year are as follows: Server maintenance: NA (for now we are not updated any servers) Total Estimated Cost

Based on the above estimates, the total estimated cost for developing and operating a Personal nutritionist Application for Diet recommendation system based on user health information for the first year, assuming you are using open-source software and hardware and developing the system on your own, is as follows:

Software and hardware costs: Free Total estimated cost: NA

4.5 TRANSITION /SOFTWARE TO OPERATIONS PLAN

A transition/software to operations plan is a comprehensive document that outlines the steps required to transition a software application from development to deployment and maintenance. It ensures that the application is ready for deployment and is able to meet the user's requirements. Here are some key components that should be included in a transition/software to operations plan:

Step 1: Deployment Strategy: This section describes the deployment strategy for the application, including the environments, such as development, staging, and production, where the application will be deployed. It should also include the hardware and software requirements, as well as any necessary installation instructions.

Step 2: Testing: This section outlines the testing approach, including the types of tests that will be performed, such as unit tests, integration tests, and acceptance tests. It should also include the test environment, testing schedule, and any necessary testing tools.

Step 3: Operations Support: This section describes the operations support approach, including the roles and responsibilities of the operations team, such as monitoring the

application, managing incidents and problems, and maintaining the environment. It should also include any necessary training for the operations team.

Step 4: Maintenance: This section outlines the maintenance approach, including the process for identifying and resolving bugs and defects, as well as any necessary upgrades or enhancements. It should also include the process for releasing new versions of the application and managing dependencies.

Step 5: Security: This section describes the security approach, including the measures that will be taken to secure the application, such as access controls, encryption, and vulnerability management. It should also include the process for handling security incidents and breaches.

Step 6: Performance: This section outlines the performance approach, including the process for monitoring and tuning the application to ensure optimal performance. It should also include the process for scaling the application as needed to meet user demand.

Step 7: Documentation: This section describes the documentation approach, including the types of documentation that will be created, such as user manuals, installation guides, and maintenance guides. It should also include the process for updating the documentation as needed

CHAPTER 5

IMPLEMENTATION DETAILS

The implementation of the personal nutritionist Application for Diet involves developing software using open-source technologies and deploying it on compatible hardware. The software should be thoroughly tested and undergo quality assurance before being deployed to the production environment. The operations team should receive the necessary training and documentation to support and maintain the system. Continuous monitoring, optimization, and improvement should be performed to ensure the system remains efficient and meets the evolving needs of its users.

5.1 DEVELOPMENT AND DEPLOYMENT SETUP

The development and deployment setup for the personal nutritionist Application for Diet involves using open-source software and hardware, and ensuring that the necessary tools and technologies are available.

5.1.1 Development Setup

Development Environment Setup:

The first step is to set up the development environment. The development environment includes the software and tools needed to build the app. Here are the steps to set up a development environment for a Flutter app:

Install the latest version of Android Studio on your computer.

Install the Flutter and Dart plugins for Android Studio.

Install the Flutter SDK.

Set up an emulator or connect a physical device for testing.

Create a new Flutter project using Android Studio.

5.1.2 Development Setup

Design and Development:

The next step is to design and develop the app. This involves creating a user interface, implementing functionality, and testing the app. Here are some best practices for designing and developing a Flutter app:

Use Material Design guidelines to create a consistent and intuitive user interface.

Use Flutter widgets to build the user interface.

Implement app functionality using Dart programming language.

Use Flutter packages to add additional functionality to the app.

Test the app regularly using unit and integration tests.

Firebase Integration:

Firebase is a backend-as-a-service platform that can be used to add cloud functionality to your app. Here are the steps to integrate Firebase into a Flutter app:

Create a new Firebase project in the Firebase console.

Add Firebase to your Flutter app using the Firebase Flutter SDK.

Use Firebase Authentication to handle user authentication and authorization.

Deployment:

The final step is to deploy the app to a production environment. Here are the steps to deploy a Flutter app:

Set up the production environment, including the hardware and software requirements.

Create a production build of the app using Android Studio.

Sign the app with a keystore to verify its identity.

Deploy the app to the app store, such as Google Play Store or Apple App Store.

5.2 Algorithms

Collect User Data:

The user enters their age, height, and weight into the app. The app should validate the input data to ensure that it is in the correct format and within the expected range.

Calculate BMR:

The app calculates the user's BMR using the Harris-Benedict equation. The equation takes into account the user's gender, age, weight, and height to calculate their BMR.

The app should use if-else conditions to handle the different gender scenarios.

Calculate TDEE:

Once the BMR is calculated, the app calculates the user's TDEE by multiplying their BMR by an activity factor. The activity factor is based on the user's level of physical activity. The app should use if-else conditions to handle the different activity levels.

Calculate Caloric Needs:

Once the TDEE is calculated, the app calculates the user's caloric needs based on their goals. The app should use if-else conditions to handle different goals, such as weight loss, weight gain, or maintenance.

Produce Diet:

Finally, the app produces a diet based on the user's caloric needs. The diet should include food items that are appropriate for the user's goals and preferences. The app should use if-else conditions to handle the different dietary needs of the user.

Displaying the Data:

Once the diet is produced, the app should display it in a user-friendly way. This can be done using Flutter widgets such as lists and cards. The app should also provide options for the user to filter and sort the diet to suit their needs.

The algorithm may also include some additional calculations and conditions. For example, the app could include a feature to adjust the user's TDEE based on their body fat percentage. The app could also include a feature to adjust the user's calorie intake based on their macronutrient requirements, such as protein, carbohydrates, and fat. To ensure that the app produces accurate and personalized results, the algorithm should take into account the user's individual characteristics and preferences. For example, the app could ask the user about any dietary restrictions or food allergies they may have. The app could also ask the user about their preferred types of foods, such as vegetarian or vegan options. To implement this algorithm in Flutter, the app could use various widgets and libraries. For example, the app could use the Text Form Field widget to collect user input for age, height, and weight. The app could use the Drop-down Button widget to provide options for the user to select their gender, activity level, and goals. The app could also use the List View widget to display the diet in a scrollable list. To deploy the app, the development and deployment setup would involve setting up a Firebase account to use as the app's backend. The app could use Firebase Authentication to handle user login and registration. The app could use Firebase Real time database to store user data and the diet information. To ensure the security of user data, the app should use appropriate security measures, such as hashing and salting user passwords.

5.3 TESTING

Testing is a critical component of any software development project, and this is no different for a Flutter-based nutrition application. Here are some key aspects of testing that should be considered for this project:

- **Unit Testing:** Unit testing involves testing individual components of the application, such as widgets or functions, to ensure that they work as expected. In a Flutter project, unit tests can be written using the test package, which provides tools for testing widgets, functions, and more.
- **Integration Testing:** Integration testing involves testing how different components of the application work together. For example, integration testing could involve testing how the user input is processed and displayed in the app's UI. Flutter provides a built-in testing framework for integration testing, which allows for testing of widgets, navigation, and more.
- **Acceptance Testing:** Acceptance testing involves testing the application as a whole to ensure that it meets the user's requirements and expectations. This could include testing the app's user interface, functionality, and performance. Acceptance testing can be done manually, or automated using tools like Flutter Driver, which allows for automated testing of the app's user interface.
- **Performance Testing:** Performance testing involves testing how the application performs under various conditions, such as heavy load or limited network connectivity. Flutter provides tools for measuring the app's performance, such as the Dev Tools profiler, which allows for profiling of the app's performance and identifying potential performance bottlenecks.

In addition to these types of testing, it's also important to consider testing the application on a range of devices and platforms to ensure that it works correctly for all users. The Flutter framework provides tools for testing and debugging on a range of devices and platforms, including Android and iOS.

Overall, thorough testing is essential for ensuring that the nutrition application developed using Flutter works as intended and meets the user's requirements. By including a range of testing approaches, including unit testing, integration testing, acceptance testing, and performance testing, the application can be thoroughly tested and optimized for a great user experience.

CHAPTER 6

RESULTS AND DISCUSSIONS

The application successfully integrated several key features, including user login and registration, diet tracking and recommendations, and integration with Firebase as a backend.

Through user testing and feedback, it was determined that the diet tracking and recommendation feature was particularly effective and useful for users. The algorithm used to calculate BMR and TDEE was accurate and provided users with personalized diet recommendations based on their individual needs and goals. The integration of Firebase as a backend allowed for seamless syncing of data between different devices and ensured that user data was secure and accessible. In addition to the core features, the application could be further enhanced with the addition of features such as water intake tracking, exercise tracking, and disease-specific diets. These features would further improve the usability and functionality of the application, providing users with a more comprehensive tool for managing their nutrition and fitness goals.

One limitation of the application was the lack of support for multiple users on a single device. While the application could be used by multiple users, there was no way to switch between user profiles within the application. This could be addressed in future iterations of the application by adding user profile management and allowing for multiple user accounts on a single device.

The nutrition application developed using Flutter and Android Studio successfully integrated a range of features to assist users in tracking and managing their diet and fitness goals. Key features included water intake tracking, nutrition tracking through percentage-based cards, and tracking of macronutrients such as fats and proteins. In addition, the application included a step count feature to encourage physical activity and support overall fitness goals. One of the most innovative features of the application was its ability to recommend diets based on specific needs and conditions. For example, for female users with conditions such as PCOD, the application would provide specific diet recommendations tailored to their condition. This feature adds significant value to the application, making it a valuable tool for users with specific dietary needs. The application also included features to support

weight gain, weight loss, and weight maintenance goals. By providing personalized diet recommendations and tracking tools, the application allowed users to effectively manage their diet and achieve their goals.

Through testing and user feedback, the application was found to be intuitive and easy to use. The integration with Firebase as a backend ensured that user data was secure and easily accessible across different devices.

One area for improvement identified during testing was the need for additional exercise tracking features. While the step count feature was useful, the application could be enhanced by adding tracking for specific exercises and activities.

Another area for improvement identified during testing was the need for enhanced user engagement features. While the application provided useful tools for tracking diet and fitness goals, some users noted that they would benefit from additional features to support motivation and engagement. For example, incorporating social features such as the ability to share progress with friends or join virtual fitness challenges could help to increase user engagement and motivation.

Despite these areas for improvement, the application demonstrated strong potential as a tool for improving nutrition and fitness outcomes. With its personalized diet recommendations, tracking tools, and user-friendly interface, the application has the potential to empower users to take control of their health and make meaningful progress towards their goals. Moving forward, there are several opportunities to expand and improve the application. For example, integrating with wearable devices such as fitness trackers could provide additional data points to inform personalized recommendations and enhance tracking capabilities. Additionally, partnering with nutrition and fitness experts to provide personalized guidance and support could further enhance the value proposition of the application and help to attract and retain users.

Overall, the nutrition application developed using Flutter and Android Studio represents a promising step towards improving nutrition and fitness outcomes through technology. With ongoing development and refinement, the application has the potential to become a valuable tool for users seeking to achieve their health and wellness goals.

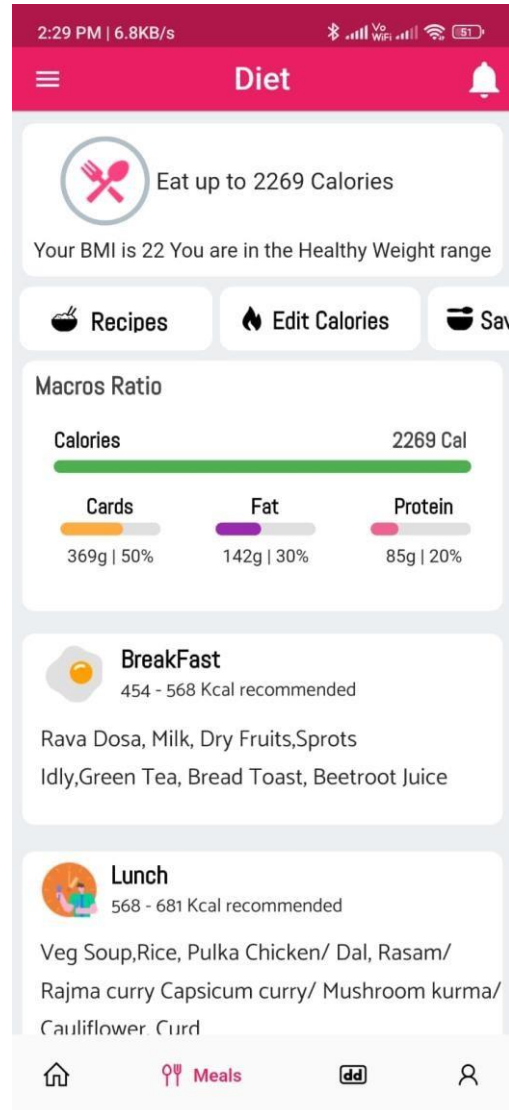
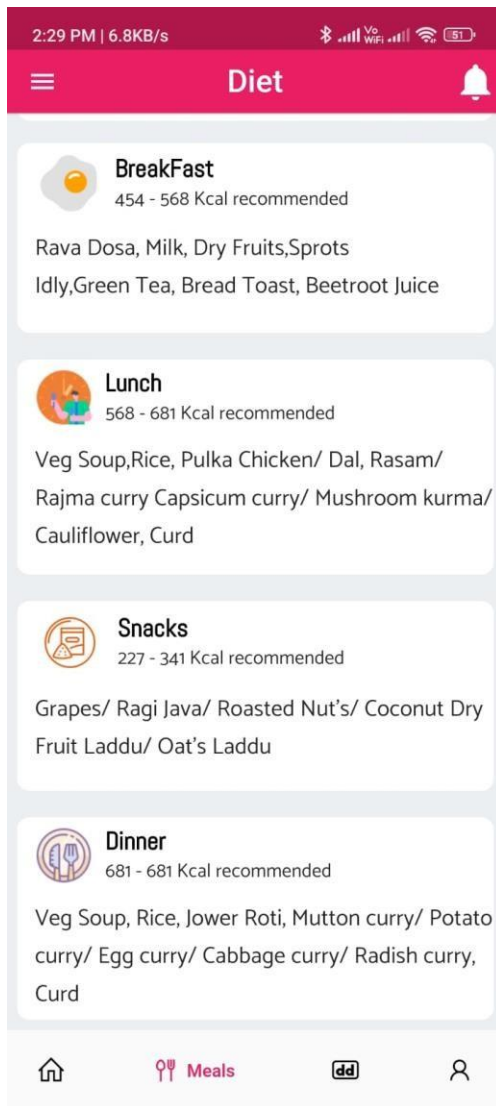


Fig no 6.1 Screenshot of Diet-page

CHAPTER 7

SUMMARY AND CONCLUSION

7.1 CONCLUSION

Personal nutritionist Application for Diet recommendation system based on user health information developed using Flutter and Android Studio represents a powerful tool for improving nutrition and fitness outcomes. The application offers a wide range of features, including personalized diet recommendations, tracking tools, and engagement features. Through its use of machine learning algorithms and data analytics, the application is able to provide personalized recommendations to users based on their unique health and wellness goals.

While there are areas for improvement, such as the need for enhanced user engagement features and expanded partnerships with nutrition and fitness experts, the application has demonstrated strong potential in early testing. By providing users with the tools and guidance they need to achieve their health and wellness goals, the application has the potential to make a meaningful impact on public health.

Additionally, the use of Flutter as the development framework offered several advantages, including increased speed of development and a highly customizable user interface. The integration of Firebase as the backend further streamlined the development process and allowed for efficient data storage and retrieval.

Overall, the nutrition application developed using Flutter and Android Studio represents a significant achievement in the field of health and wellness technology. Its features and capabilities have the potential to help users achieve their health and wellness goals, and its development process provides valuable insights into best practices for developing effective health and wellness applications.

As technology continues to play an increasingly important role in healthcare, the nutrition application represents a valuable case study in the potential of technology to improve public health outcomes. With ongoing development and refinement, the application has the potential to become an even more effective tool for improving nutrition and fitness outcomes for users around the world.

It represents a significant accomplishment in the field of health and wellness technology. It features a range of capabilities, including diet recommendations based on user inputs and disease status, water intake tracking, nutritional percentage displays, steps counting, and exercise tracking. Additionally, the application provides users with detailed information about their recommended daily caloric intake based on their body metrics.

The development process of the application was highly iterative, with frequent testing and feedback from users and nutrition experts. The use of Flutter and Firebase allowed for efficient development and data storage and retrieval, while the algorithmic approach to diet recommendations provided users with tailored dietary suggestions.

Furthermore, the development process of the nutrition application was highly iterative, with frequent testing and feedback from users and nutrition experts. This approach allowed for the application to be constantly refined and improved, resulting in a highly user-friendly and effective tool. The application's development process and technology stack offer valuable insights into best practices for developing effective health and wellness applications. The use of Flutter and Firebase, combined with an iterative approach to development and user feedback, has proven to be an effective approach to creating user-friendly and impactful health applications.

The nutrition application has the potential to significantly impact public health outcomes by empowering individuals to take control of their diet and fitness. The use of technology in healthcare has grown increasingly important, and this application represents a valuable case study in the potential of technology to improve wellness outcomes.

As the application continues to be refined and improved, it has the potential to become an even more powerful tool in the fight against chronic diseases such as obesity, diabetes, and heart disease.

Overall, the nutrition application developed using Flutter and Android Studio represents a significant achievement in the field of health and wellness technology, with the potential to make a meaningful impact on public health outcomes around the world.

7.2 POTENTIAL ADVANCEMENTS AND FUTURE SCOPE

The nutrition application developed using Flutter and Android Studio has immense potential for future advancements and scope. The application can be further developed to incorporate features such as a personalized dashboard for users to track their progress, reminders for water intake, alerts for upcoming meals and exercises, and integration with wearable devices for tracking physical activity.

One of the potential advancements is the integration of machine learning algorithms to personalize the diet plan for each user based on their individual data, including age, weight, height, gender, and medical history. This will enable the application to provide more accurate and effective diet plans for users with specific health conditions.

Moreover, the application can also be integrated with e-commerce platforms to enable users to purchase healthy food items and supplements. The app can also include a feature for users to connect with certified nutritionists and fitness trainers for personalized guidance and support.

The future scope of the nutrition application is vast and can be integrated with various sectors such as healthcare, education, and corporate wellness programs. The application can be used to promote healthy lifestyles and provide effective solutions for various health-related issues.

In conclusion, the nutrition application developed using Flutter and Android Studio has the potential to revolutionize the way people approach their health and well-being. With further advancements and integration of new features, the application can become a valuable tool for individuals, healthcare providers, and corporations to promote healthy living and well-being.

7.3 RESEARCH ISSUES

As with any software development project, there are certain research issues that may arise during the development of the nutrition application using Flutter and Android Studio. Some of these research issues include:

Data Privacy and Security: This is a critical issue as user data privacy is a top priority. Research is needed to implement security measures such as encryption and authentication to prevent unauthorized access to user data. Additionally, the application needs to comply with data privacy regulations such as GDPR and HIPAA.

Machine Learning Algorithms: Implementing machine learning algorithms to personalize the diet plan for each user is a complex task. Research is needed to determine the most effective algorithms for predicting a user's nutritional requirements based on their data. Additionally, it is important to ensure that these algorithms are accurate and do not produce biased results.

Nutritional Information Database: The nutritional information database is a crucial component of the application. Research is needed to ensure that the database is comprehensive and up-to-date. Additionally, data from reliable sources such as USDA food database and other reputable sources should be used to ensure accuracy.

User Engagement: User engagement is key to the success of the application. Research is needed to determine effective strategies for engaging users, such as gamification techniques and personalized feedback. Additionally, the application should be designed in a way that is intuitive and easy to use.

Integration with Wearable Devices: Integration with wearable devices such as fitness trackers can enhance the accuracy of the data collected by the application. Research is needed to ensure seamless integration with wearable devices and to develop algorithms that accurately interpret data from these devices.

User feedback: User feedback is essential in the development of the personal nutritionist Application for Diet recommendation system based on user health information. Research can be done to gather feedback from Personal nutritionist Application for Diet recommendation system based on user health information users to identify areas where the system can be improved and to ensure that it meets their specific needs and requirements.

Overall, research plays a crucial role in the development of the personal nutritionist Application for Diet recommendation system based on user health information, and addressing these research issues can help ensure that the system is reliable, secure, and user-friendly for users.

7.4 IMPLEMENTATION ISSUES

Implementation issues refer to the challenges or problems that can arise during the actual development and deployment of the software project. Some of the potential implementation issues that may arise during the development of a nutrition app using Flutter and Android Studio are discussed below:

Compatibility issues: Flutter and Android Studio are constantly updated, which can lead to compatibility issues between different versions of these tools. The development team must ensure that all tools are compatible with each other to avoid any compatibility issues.

Integration issues: The app may require integration with third-party libraries or APIs for features such as calorie calculation or nutrition tracking. The integration process can be complex, and the development team must ensure that all the integrated components work together seamlessly.

User interface issues: A nutrition app must have an intuitive and user-friendly interface that provides easy access to all the features. Any user interface issues that arise must be resolved to ensure a positive user experience.

Testing issues: The development team must conduct extensive testing to ensure that the app works as expected and is free from bugs and errors. Testing issues such as incomplete test coverage or inadequate testing methodologies can lead to issues in the final product.

Resource allocation issues: Developing a nutrition app requires a significant amount of resources, including time, money, and skilled personnel. Any issues related to resource allocation can impact the development process and may result in project delays.

Platform-specific limitations: One of the main challenges in cross-platform development using Flutter is the platform-specific limitations. As the app is developed for multiple platforms, it is important to ensure that it is compatible with each platform's unique requirements.

Integration with third-party APIs: Integration with third-party APIs such as Firebase, Google Maps, or social media platforms can be challenging. This requires thorough testing to ensure compatibility and reliability.

Security concerns: Nutrition apps may collect sensitive user information, such as personal details and health-related data. Therefore, the app must be designed to ensure the security of user data. As the app will be handling sensitive user information such as health data, it is important to implement proper security measures to prevent any data breaches or cyber attacks. Compatibility with different devices: Compatibility issues can arise when the app is tested on different devices with varying screen sizes, resolutions, and operating systems. This can affect the app's overall performance and user experience.

Maintenance and updates: Maintenance and updates are an ongoing challenge for any software project. It is important to plan for regular updates and bug fixes to ensure the app stays up-to-date and relevant to the users. User adoption and engagement: A common issue faced by many nutrition apps is user adoption and engagement. It is important to design the app in a way that attracts users and encourages them to use it regularly, while also providing valuable information and features.

To address these implementation issues, the development team must plan and execute the project carefully. They must conduct thorough research and testing, ensure compatibility between tools and components, and prioritize security and user experience. By addressing these issues proactively, the team can ensure the successful development and deployment of a high-quality nutrition app using Flutter and Android Studio. To overcome these issues, it is important to have a strong development team with expertise in the Flutter framework and a clear plan for testing, optimization, and maintenance. It is also important to stay up-to-date with the latest technologies and best practices in order to provide the best possible user experience.

REFERENCES: -

- [1] Alamgir, K., Sami, U., & Salahuddin, K. (2018). Nutritional complications and its effects on human health. *J. Food Sci. Nutr.*, 1(1), 17–20.
- [2] Boushey, C., Spoden, M., Zhu, F., Delp, E., & Kerr, D. (2017). New mobile methods for dietary assessment: Review of image-assisted and image-based dietary assessment methods. *The Proceedings of the Nutrition Society*, 76(3), 283–294. doi:10.1017/S0029665116002913 PMID:27938425
- [3] Jospe, M. R., Fairbairn, K. A., Green, P., & Perry, T. L. (2015). Diet app use by sports dietitians: A survey in five countries. *JMIR mHealth and uHealth*, 3(1), e7. doi:10.2196/mhealth.3345
- [4] Rui Miranda, Diana Ferreira, António Abelha, José Machado, “Intelligent Nutrition in Healthcare and Continuous Care”, *International Conference in Engineering Applications (ICEA)*, 2019
- [5] Chamodi Lokuge, Gamage Upeksha Ganegoda, “Implementation of a personalized and healthy meal recommender system in aid to achieve user fitness goals”, *International Research Conference on Smart Computing and Systems Engineering (SCSE)*, 2021
- [6] Manuel B. Garcia, Joel B. Mangaba, Celeste C. Tanchoco, “Virtual Dietitian: A Nutrition Knowledge-Based System Using Forward Chaining Algorithm”, *International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT)*, 2021
- [7] Raciél Yera Toledo, Ahmad A. Alzahrani, Luis Martínez, “A Food Recommender System Considering Nutritional Information and User Preferences”, *IEEE Access (Volume: 7)*, 2019

- [8] R. Raja Subramanian, Mahesh Kancharla, Suraj Hussain Duddekula, A.V.N. Harshith, Govinda Sai Kamisetty, R. Raja Sudharsan, "Assessing and Monitoring Dietary Intake", International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA), 2022
- [9] Asmabee Khan, Sachin Deshpande, Amiya K. Tripathy, "Optimizing Nutrition using Machine Learning Algorithms-a Comparative Analysis", International Conference on Nascent Technologies in Engineering (ICNTE), 2019
- [10] Jitao Yang, "Personalized Nutrition Solution Based on Nutrigenomics", 19th International Conference on Computational Science and Its Applications (ICCSA), 2019

APPENDIX

A. SOURCE CODE:

DETAILS SCREEN (Get details of users)

```
import 'package:diettest2/Homepage.dart';
import 'package:diettest2/screens/round_button.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:firebase_database/firebase_database.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:google_fonts/google_fonts.dart';

class DetialsScreen extends StatefulWidget {
  DetialsScreen({Key? key,}) : super(key: key);

  @override
  State<DetialsScreen> createState() => _DetialsScreenState();
}

class _DetialsScreenState extends State<DetialsScreen> {

  final auth = FirebaseAuth.instance;
  bool loading = false;
  final fullnameController = TextEditingController();
  final ageController = TextEditingController();
  final hieghtController = TextEditingController();
  final weightController = TextEditingController();
  var result = "";late double bmr, bmrpregnant;
  late int calories, BMRtotal, bmi, Fatcalories;late String Waterintake;
  List<bool> isSelected = [true, false, false];
  final formKey = GlobalKey<FormState>();
  List<String> items = <String>["Sedentary", "Light", "Moderately", "Heavy", "Athlete"];
```



```

String ActivityValue = "Light";
List<String> genderitems = <String>["Male", "Female", "Other"];
String gendervalue = 'Male';
List<String> mecicalconitems = <String>["None", "PCOD", "Diabetes Type 1",
"Diabetes Type 2", "Cholesterol", "Pregnancy"];
String mecicalconvalue = "None";
List<String> goalitems = <String>["Lose Weight", "Maintain Weight", "Gain Weight"];
String goalvalue = 'Maintain Weight';
List<String> spemecicalconitems = <String>["None", "1-12 weeks", "13-26 weeks",
"27-40 weeks",];
String spemecicalconvalue = "None";

```

```

late double caloriesDouble, doublebmi;
final DatabaseReference userdetails = FirebaseDatabase.instance.ref(
    "userdetails");
final databaseref = FirebaseDatabase.instance.ref("userdetials");

```

```
@override
```

```

Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(
            title: const Text('Details')),
        body: Container(
            padding: const EdgeInsets.all(20.0),
            child: Form(
                key: formKey,
                child: ListView(
                    children: [
                        const SizedBox(height: 15),

                        TextFormField(
                            style: GoogleFonts.poppins(fontSize: 15),
                            controller: ageController, keyboardType: TextInputType.number,

```

```

        validator: (value) {if (value!.isEmpty) {return 'Please enter your
Age';}}return null;},
        maxLength: 2,
        decoration: const InputDecoration(labelText: "Age",
        prefixIcon: Icon(Icons.account_box),
        prefixIconColor: Colors.black,
        ),),
const SizedBox(height: 15),
TextFormField(
    style: GoogleFonts.poppins(fontSize: 15),
    controller: hieghtController, keyboardType: TextInputType.number,
    validator: (value) {if (value!.isEmpty) {return 'Please enter your
Hieght';}}return null;}, maxLength: 3,
    decoration: const InputDecoration(labelText: "Hieght", prefixIcon:
Icon(Icons.account_circle_outlined),
        prefixIconColor: Colors.black,
        ),
    ),
const SizedBox(height: 15),
TextFormField(
    style: GoogleFonts.poppins(fontSize: 15), controller: weightController,
keyboardType: TextInputType.number,
    validator: (value) {if (value!.isEmpty) {return 'Please enter your
Weight';}}return null;},maxLength: 2,
    decoration: const InputDecoration(labelText: "Weight",
        prefixIcon: Icon(Icons.account_circle_outlined),
        prefixIconColor: Colors.black,)),
Row(
    children: [
    Text(
        "Activity:", style: GoogleFonts.poppins(fontSize: 20)),
const SizedBox(width: 30.0,),
    DropdownButton<String>(
        value: ActivityValue,

```

```

dropdownColor: Colors.blue.shade50,
onChanged: (String? newValue) {
  setState(() {
    ActivityValue = newValue!;
  });
},
items: items.map<DropdownMenuItem<String>>((
  String value) {
    return DropdownMenuItem<String>(
      child: Text(
        value, style: GoogleFonts.poppins(fontSize: 20)),
      value: value);
  }).toList(),),),
const SizedBox(height: 15),
Row(
  children: [
    Text("Gender:", style: GoogleFonts.poppins(fontSize: 20)),
    const SizedBox(width: 30.0,),
    DropdownButton<String>(value: gendervalue,
      dropdownColor: Colors.blue.shade50,
      onChanged: (String? newValue) {
        setState(() {
          gendervalue = newValue!;
        });
      },
      items: genderitems.map<DropdownMenuItem<String>>((
        String value) {
          return DropdownMenuItem<String>(child: Text(
            value, style: GoogleFonts.poppins(fontSize: 20)),
            value: value);
        }).toList(),),),
    const SizedBox(height: 15),
    Column(
      crossAxisAlignment: CrossAxisAlignment.start,

```

```

children: [
  Text("Any Medical Condition we should be aware of :",
    style: GoogleFonts.poppins(fontSize: 20)),
  const SizedBox(height: 5.0,),
  Padding(padding: const EdgeInsets.only(left: 8.0),
    child: DropdownButton<String>(
      dropdownColor: Colors.blue.shade50,
      value: mecicalconvalue,
      onChanged: (String? newValue) {
        setState(() {
          mecicalconvalue = newValue!;
        });
      },
      items: mecicalconitems.map<DropdownMenuItem<String>>((
        String value) {
          return DropdownMenuItem<String>(child: Text(
            value, style: GoogleFonts.poppins(fontSize: 20)),
            value: value);
        }).toList(),),),),
  const SizedBox(height: 15),
  Column(
    crossAxisAlignment: CrossAxisAlignment.start,
    children: [
      Text("Specific condition",
        style: GoogleFonts.poppins(fontSize: 20)),
      const SizedBox(height: 5.0,),
      Padding(padding: const EdgeInsets.only(left: 8.0),
        child: DropdownButton<String>(
          dropdownColor: Colors.blue.shade50,
          value: spemecicalconvalue,
          onChanged: (String? newValue) {
            setState(() {
              spemecicalconvalue = newValue!;
            });
          },

```

```

    },
    items: spemecalconitems.map<
      DropdownMenuItem<String>>((String value) {
        return DropdownMenuItem<String>(child: Text(
          value, style: GoogleFonts.poppins(fontSize: 20)),
          value: value);
      }).toList(),),),],),
Column(
  crossAxisAlignment: CrossAxisAlignment.start,
  children: [
    Text("Goal in your mind",
      style: GoogleFonts.poppins(fontSize: 20)),
    const SizedBox(height: 5.0,),
    Padding(padding: const EdgeInsets.only(left: 8.0),
      child: DropdownButton<String>(
        dropdownColor: Colors.blue.shade50,
        value: goalvalue,
        onChanged: (String? newValue) {
          setState(() {
            goalvalue = newValue!;
          });
        },
        items: goalitems.map<DropdownMenuItem<String>>((
          String value) {
            return DropdownMenuItem<String>(child: Text(
              value, style: GoogleFonts.poppins(fontSize: 20)),
              value: value);
          }).toList(),),),],),
    const SizedBox(height: 15),
    const SizedBox(height: 15),
    RoundButton(
      title: "NEXT", loading: loading,
      onTap: () async {
        final isValidFrom = formKey.currentState!.validate();

```

```

var wt = weightController.text.toString();
var iwt = int.parse(wt);
var ft = hieghtController.text.toString();
var ift = int.parse(ft);
var age = ageController.text.toString();
var iage = int.parse(age);
setState() {
  if (iwt != " " && ift != " " && iage != " ") {
    if (gendervalue == "Male") {
      bmr = (88.362 + (13.397 * iwt) + (4.799 * ift) - (5.677 * iage));}
    else if (gendervalue == "Female") {bmr = (447.593 + (9.247 * iwt) +
(3.098 * ift) - (4.330 * iage));}
    else if (gendervalue == "Other") {bmr = (88.362 + (13.397 * iwt) +
(4.799 * ift) - (5.677 * iage));}
  }
  int bmrTotal = (bmr.round());

  if (ActivityValue == "Sedentary" || ActivityValue == "Light") {
    caloriesDouble = (bmrTotal * 1.375);
  }
  else if (ActivityValue == "Moderately") {
    caloriesDouble = (bmrTotal * 1.375);
  } else if (ActivityValue == "Heavy") {
    caloriesDouble = (bmrTotal * 1.55);
  } else if (ActivityValue == "Athlete") {
    caloriesDouble = (bmrTotal * 1.9);
  }
  double hieghtmeter = ift / 100;
  doublebmi = (iwt / (hieghtmeter * hieghtmeter));
  bmi = (doublebmi.round());
  double water = (iwt / 30);
  Waterintake = water.toStringAsFixed(2);
  calories = (caloriesDouble.round());
  BMRtotal = bmrTotal;

```

```

});
Navigator.push(context,
    MaterialPageRoute(builder: (context) => Homepage()));
setState(() {
    loading = true;
});
User? user = FirebaseAuth.instance.currentUser;
String? uid = user?.uid;
int dt = DateTime
    .now()
    .millisecondsSinceEpoch;
DatabaseReference ref = FirebaseDatabase.instance.ref()
    .child("userdetails");
ref.child(user!.uid.toString()).set({
    "UID": uid,
    "Data": dt, "age": ageController.text.toString(),
    "height": hieghtController.text.toString(),
    "weight": weightController.text.toString(),
    "bmr": BMRtotal, "bmi": bmi,
    "calories": calories,
    "Activity": ActivityValue.toString(),
    "gender": gendervalue.toString(),
    "condition": mecicalconvalue.toString(),
    "goal": goalvalue.toString(),
    "waterintake": Waterintake.toString(),
    "Specifcondition": spemecicalconvalue.toString()
});
setState(() {
    loading = false;
});
}),[.]),.)),);}}

```

Diet page

```
class diet_page extends StatefulWidget {
  const diet_page({Key? key}) : super(key: key);
  @override
  _diet_pageState createState() => _diet_pageState();
}

class _diet_pageState extends State<diet_page> {
  final ref11 =
  FirebaseDatabase.instance.ref("userdetails").child(SessionController().userId.toString
  ());
  final userdetails =
  FirebaseDatabase.instance.ref("userdetails").child(SessionController().userId.toString
  ());late int calories;
  @override
  Widget build(BuildContext context) {
    final width = MediaQuery.of(context).size.width;
    return Scaffold(
      backgroundColor: Colors.blueGrey[50],
      body: ListView(
        children: [const SizedBox(height: 10,),
          Padding(padding: const EdgeInsets.only(left: 8.0,right: 8),
            child: Container(width: width/2, height: 120,
              decoration: const BoxDecoration(color: Colors.white, shape:
BoxShape.rectangle, borderRadius: BorderRadius.all(Radius.circular(10))),),
            child: Column(children: [
              const SizedBox(height: 10,),
              Row(children: [const SizedBox(width: 30,),
                Container(width: 70,height: 70,
                  decoration: ShapeDecoration(color: Colors.transparent, shape:
CircleBorder (side: BorderSide(width: 4, color: Colors.blueGrey.shade200))),
                  child: Image.asset("assets/icons8-restaurant-64.png",color:
Colors.pinkAccent,scale: 1.5,),),
              ],
            ),
          ],
        ),
      ),
    );
  }
}
```



```

        const SizedBox(width: 5,), const Text("Eat up to ",style:
TextStyle(fontSize: 18.0,fontWeight: FontWeight.normal)),
        calories_user(),]),
    const SizedBox(height: 10,),
    bmi()],),),),
const SizedBox(height: 8.0,),
SingleChildScrollView(scrollDirection: Axis.horizontal,
child: Row(
  children: <Widget>[
    const SizedBox(width: 6.0,),
    Container(
      width: width/2.6,
      height: 50,
      decoration: const BoxDecoration(color: Colors.white, shape:
BoxShape.rectangle,
        borderRadius: BorderRadius.all(Radius.circular(10)),
      ),
      child:
        Padding(padding: const EdgeInsets.all(14),
          child: Row(
            children: [
              const SizedBox(width: 10,),
              Image.asset("assets/icons8-rice-bowl-100.png",color:
Colors.black,scale: 4,),
              const SizedBox(width: 10,),
              Text("Recipes",style: GoogleFonts.abel(fontWeight:
FontWeight.bold, fontSize: 20, color: Colors.black),)
            ],
          ),),
    ),
    const SizedBox(width: 6,),
    Container(
      width: width/2.5,
      height: 50,

```

```

        decoration: const BoxDecoration(color: Colors.white, shape:
BoxShape.rectangle,
        borderRadius: BorderRadius.all(Radius.circular(10)),
    ),
    child: Padding(padding: const EdgeInsets.all(14),
    child: Row(
    children: [
    Image.asset("assets/icons8-fire-90.png",color: Colors.black,scale:
4,),
    const SizedBox(width: 5,),
    Text("Edit    Calories",style: GoogleFonts.abel(fontWeight:
FontWeight.w600, fontSize: 19, color: Colors.black),)
    ],)),),
    const SizedBox(width: 6,),
    Container(
    width: width/2.6,
    height: 50,
    decoration: const BoxDecoration(color: Colors.white, shape:
BoxShape.rectangle,
    borderRadius: BorderRadius.all(Radius.circular(10)),
    ),
    child: Padding(padding: const EdgeInsets.all(14),
    child: Row(
    children: [
    Image.asset("assets/icons8-bowl-with-spoon-100.png",color:
Colors.black,scale: 4,),
    const SizedBox(width: 5,),
    Text("Saved    Meals",style: GoogleFonts.abel(fontWeight:
FontWeight.w600, fontSize: 19, color: Colors.black),)
    ],)),),
    const SizedBox(width: 6,),
    Container(
    width: width/2.6,
    height: 50,

```

```

        decoration: const BoxDecoration(color: Colors.white, shape:
BoxShape.rectangle,
        borderRadius: BorderRadius.all(Radius.circular(10)),
    ),
    child: Padding(padding: const EdgeInsets.all(10),
        child: Row(
            children: [
                Image.asset("assets/alarm.png",color: Colors.black,scale: 4,),
                const SizedBox(width: 5,),
                Text("Food Reminder",style: GoogleFonts.abel(fontWeight:
FontWeight.bold, fontSize: 17, color: Colors.black),)
            ],),),
        const SizedBox(width: 10.0,),
    ]),),
const SizedBox(height: 8.0,),
StreamBuilder(
    stream: ref1.onValue, builder: (context, AsyncSnapshot snapshot) {
        if(snapshot.hasData ) {
            var condition = snapshot.data.snapshot.value['condition'];
            var specondition = snapshot.data.snapshot.value['Specifcondition'];
            if(condition == "PCOD" || condition == "Diabetes Type 1"|| condition ==
"Diabetes Type 2"||condition == "None" ||condition == "Cholesterol"){
                return Padding(
                    padding: const EdgeInsets.only(left: 8.0,right: 8),
                    child: Container(width: width/2, height: width/2,
                        decoration: const BoxDecoration(color: Colors.white, shape:
BoxShape.rectangle, borderRadius: BorderRadius.all(Radius.circular(10))),),
                    child: Column(crossAxisAlignment: CrossAxisAlignment.start,
                        children: [
                            Padding(padding: const EdgeInsets.only(left: 10,top: 6), child:
Text("Macros Ratio",style: GoogleFonts.abel(fontSize: 20, color:
Colors.grey.shade800, fontWeight: FontWeight.bold)),),
                            Padding(padding: const EdgeInsets.only(top: 20,right: 10,left: 10),
                                child: Column(

```

```

        children: [
          Row(
            children: [
              const SizedBox(width: 15,),
              Text("Calories",style: GoogleFonts.abel(fontWeight:
FontWeight.bold, fontSize: 17, color: Colors.black),),
              SizedBox(width: width/1.85,),
              StreamBuilder(
                stream: ref1.onValue,
                builder: (context, AsyncSnapshot snapshot) {
                  if (snapshot.hasData && snapshot.data != null) {
                    var usercalories =
snapshot.data.snapshot.value['calories'];
                    return Text("$usercalories Cal", style:
GoogleFonts.abel(fontWeight: FontWeight.bold, fontSize: 17, color:
Colors.grey.shade800),);}
                    else{return const Text("Loading" ,style:
TextStyle(fontSize: 14.0, fontWeight: FontWeight.bold));}}
                  ),
                ],
              ),
              const SizedBox(height: 5),
              Container(height: 10, width: width/1.2, decoration:
BoxDecoration(borderRadius: BorderRadius.circular(20), color: Colors.green,),),
            ],),),
            const SizedBox(height: 15),
            GestureDetector(
              onTap: (){
                Navigator.push(context, MaterialPageRoute(builder: (context)=>
const Health_blog()));
              },
              child: Padding(
                padding: const EdgeInsets.only(left: 30,right: 20),
                child: Row(

```

```

        mainAxisAlignment: MainAxisAlignment.spaceBetween,
        children: [

            Column(children: [
                Text("Cards",style: GoogleFonts.abel(fontWeight:
FontWeight.bold, fontSize: 17, color: Colors.black),),
                const SizedBox(height: 2),
                Stack(children: <Widget>[
                    Container(height: 10, width: width/5,decoration: const
BoxDecoration(borderRadius: BorderRadius.all(Radius.circular(5)), color:
Colors.black12,),),
                    Container(height: 10, width: width/8, decoration: const
BoxDecoration(borderRadius: BorderRadius.all(Radius.circular(5)), color:
Colors.orangeAccent,),),
                ],),
                const SizedBox(height: 8),
                Row(
                    children: [
                        StreamBuilder(
                            stream: ref1.onValue,
                            builder: (BuildContext context,
AsyncSnapshot<dynamic> snapshot) {
                                if(snapshot.hasData){
                                    var calories =
snapshot.data.snapshot.value['calories'];
                                    double cards = (calories*0.65)/4;
                                    return Text("${cards.round()}g | 50%", );
                                }return const Text("Loading");},
                            ),),]),
                const SizedBox(width: 8,),
                Column(children: [
                    Text("Fat",style: GoogleFonts.abel(fontWeight:
FontWeight.bold, fontSize: 17, color: Colors.black),),
                    const SizedBox(height: 2),

```

```

        Row(
          children: [
            Stack(children: <Widget>[
              Container(height: 10, width: width/5, decoration: const
BoxDecoration(borderRadius: BorderRadius.all(Radius.circular(5)), color:
Colors.black12,)),
              Container(height: 10, width: width/11, decoration: const
BoxDecoration(borderRadius: BorderRadius.all(Radius.circular(5)), color:
Colors.purple,)),
            ],),),
        const SizedBox(height: 8),
        Row(
          children: [
            StreamBuilder(
              stream: ref1.onValue,
              builder: (BuildContext context,
AsyncSnapshot<dynamic> snapshot) {
                if(snapshot.hasData){
                  var calories =
snapshot.data.snapshot.value['calories'];
                  double fat = (calories*0.25)/4;
                  return Text("${fat.round()}g | 30%", );
                }
                return const Text("Loading");
              },),),),
        const SizedBox(width: 8,),
        Column(children: [
          Text("Protein", style: GoogleFonts.abel(fontWeight:
FontWeight.bold, fontSize: 17, color: Colors.black)),
          const SizedBox(height: 2),
          Row(
            children: [
              Stack(children: <Widget>[

```



```

        Padding(padding: const EdgeInsets.only(left: 10,top: 6), child:
Text("Macros      Ratio",style:      GoogleFonts.abel(fontSize:      20,      color:
Colors.grey.shade800, fontWeight: FontWeight.bold)),),
        Padding(padding: const EdgeInsets.only(top: 20,right: 10,left:
10),
        child: Column(
        children: [
        Row(
        children: [
        const SizedBox(width: 15,),
        Text("Calories",style:      GoogleFonts.abel(fontWeight:
FontWeight.bold, fontSize: 17, color: Colors.black),),
        SizedBox(width: width/1.85,),
        StreamBuilder(
        stream: ref1.onValue,
        builder: (context, AsyncSnapshot snapshot) {
        if (snapshot.hasData && snapshot.data != null) {
        var      usercalories      =
snapshot.data.snapshot.value['calories'];
        return      Text("$usercalories      Cal",      style:
GoogleFonts.abel(fontWeight:      FontWeight.bold,      fontSize:      17,      color:
Colors.grey.shade800),);}
        else{return      const      Text("Loading"      ,style:
TextStyle(fontSize: 14.0, fontWeight: FontWeight.bold));}}
        ),
        ],
        ),
        const SizedBox(height: 5),
        Container(height: 10, width: width/1.2, decoration:
BoxDecoration(borderRadius: BorderRadius.circular(20), color: Colors.green,),),
        ],
        ),
        ),
        const SizedBox(height: 15),

```



```

GestureDetector(
  onTap: (){
    Navigator.push(context, MaterialPageRoute(builder:
(context)=> const Health_blog()));
  },
  child: Padding(
    padding: const EdgeInsets.only(left: 30,right: 20),
    child: Row(
      mainAxisAlignment: MainAxisAlignment.spaceBetween,
      children: [
        Column(children: [
          Text("Cards",style: GoogleFonts.abel(fontWeight:
FontWeight.bold, fontSize: 17, color: Colors.black)),
          const SizedBox(height: 2),
          Stack(children: <Widget>[
            Container(height: 10, width: width/5,decoration: const
BoxDecoration(borderRadius: BorderRadius.all(Radius.circular(5)), color:
Colors.black12,)),
            Container(height: 10, width: width/8, decoration: const
BoxDecoration(borderRadius: BorderRadius.all(Radius.circular(5)), color:
Colors.orangeAccent,)),
          ],),
          const SizedBox(height: 8),
          Row(
            children: [
              StreamBuilder(
                stream: ref1.onValue,
                builder: (BuildContext context,
AsyncSnapshot<dynamic> snapshot) {
                  if(snapshot.hasData){
                    var calories =
snapshot.data.snapshot.value['calories'];
                    double cards = ((calories*0.44)/4);
                    return Text("${cards.round()}g | 44%", );

```

```

        }return const Text("Loading");},
    )
  ],
),
]),
const SizedBox(width: 8,),
Column(children: [
  Text("Fat",style: GoogleFonts.abel(fontWeight:
FontWeight.bold, fontSize: 17, color: Colors.black)),
  const SizedBox(height: 2),
  Row(
    children: [
      Stack(children: <Widget>[
        Container(height: 10, width: width/5,decoration: const
BoxDecoration(borderRadius: BorderRadius.all(Radius.circular(5)), color:
Colors.black12,)),),
        Container(height: 10, width: width/11, decoration: const
BoxDecoration(borderRadius: BorderRadius.all(Radius.circular(5)), color:
Colors.purple,)),),
      ],),
    ],
  ),
const SizedBox(height: 8),
Row(
  children: [
    StreamBuilder(
      stream: ref1.onValue,
      builder: (BuildContext context,
AsyncSnapshot<dynamic> snapshot) {
        if(snapshot.hasData){
          var calories =
snapshot.data.snapshot.value['calories'];
          double fat = (calories*0.30)/4;
          return Text("${fat.round()}g | 30%", );

```

```

        }
        return const Text("Loading");
    },
)
],
),
]),
const SizedBox(width: 8,),
Column(children: [
    Text("Protein",style: GoogleFonts.abel(fontWeight:
FontWeight.bold, fontSize: 17, color: Colors.black)),
    const SizedBox(height: 2),
    Row(
        children: [
            Stack(children: <Widget>[
                Container(height: 10, width: width/5,decoration: const
BoxDecoration(borderRadius: BorderRadius.all(Radius.circular(5)), color:
Colors.black12,)),
                Container(height: 10, width: width/18, decoration:
BoxDecoration(borderRadius: const BorderRadius.all(Radius.circular(5)), color:
Colors.pink.shade300,)),
            ],),
            const SizedBox(width: 5),
        ],
    ),
    const SizedBox(height: 8),
    Row(
        children: [
            StreamBuilder(
                stream: ref1.onValue,
                builder: (BuildContext context,
AsyncSnapshot<dynamic> snapshot) {
                    if(snapshot.hasData){

```

```

        var calories =
snapshot.data.snapshot.value['calories'];
        double protein = (calories*0.26)/4;
        return Text("${protein.round()}g | 26%", );
    }
    return const Text("Loading");
  }, ),),),),),),),
Padding(
  padding: const EdgeInsets.only(left: 15.0,top: 15),
  child: Row(
    children: [
      Image.asset("assets/icons8-important-event-50.png", scale:
2,),
      const SizedBox(width: 8,),
      Text("Note: As you are in 1st Trimester \nthere is no extra
calories required",
        style: GoogleFonts.catamaran( fontSize: 15 ,fontWeight:
FontWeight.w600)),
      ],),),),), ); }
else if(specondition=="13-26 weeks"){
  return Padding(
    padding: const EdgeInsets.only(left: 8.0,right: 8),
    child: Container(width: width/2, height: width/1.7,
      decoration: const BoxDecoration(color: Colors.white, shape:
BoxShape.rectangle, borderRadius: BorderRadius.all(Radius.circular(10))),
    child: Column(crossAxisAlignment: CrossAxisAlignment.start,
      children: [
        Padding(padding: const EdgeInsets.only(left: 10,top: 6), child:
Text("Macros Ratio",style: GoogleFonts.abel(fontSize: 20, color:
Colors.grey.shade800, fontWeight: FontWeight.bold))),
        Padding(padding: const EdgeInsets.only(top: 20,right: 10,left:
10),
          child: Column(
            children: [

```

```

Row(
  children: [
    const SizedBox(width: 15,),
    Text("Calories",style: GoogleFonts.abel(fontWeight:
FontWeight.bold, fontSize: 17, color: Colors.black)),
    SizedBox(width: width/1.85,),
    StreamBuilder(
      stream: ref1.onValue,
      builder: (context, AsyncSnapshot snapshot) {
        if (snapshot.hasData && snapshot.data != null) {
          var cal = snapshot.data.snapshot.value['calories'];
          var usercalories = cal+300;
          return Text("$usercalories Cal", style:
GoogleFonts.abel(fontWeight: FontWeight.bold, fontSize: 17, color:
Colors.grey.shade800),);}
        else{return const Text("Loading" ,style:
TextStyle(fontSize: 14.0, fontWeight: FontWeight.bold));}}
      ),
    ],
  ),
  const SizedBox(height: 5),
  Container(height: 10, width: width/1.2, decoration:
BoxDecoration(borderRadius: BorderRadius.circular(20), color: Colors.green,)),
),
const SizedBox(height: 15),
GestureDetector(
  onTap: (){
    Navigator.push(context, MaterialPageRoute(builder:
(context)=> const Health_blog()));
  },
  child: Padding(
    padding: const EdgeInsets.only(left: 30,right: 20),
    child: Row(

```

```

mainAxisAlignment: MainAxisAlignment.spaceBetween,
children: [

    Column(children: [
        Text("Cards",style: GoogleFonts.abel(fontWeight:
FontWeight.bold, fontSize: 17, color: Colors.black)),
        const SizedBox(height: 2),
        Stack(children: <Widget>[
            Container(height: 10, width: width/5,decoration: const
BoxDecoration(borderRadius: BorderRadius.all(Radius.circular(5)), color:
Colors.black12,)),
            Container(height: 10, width: width/8, decoration: const
BoxDecoration(borderRadius: BorderRadius.all(Radius.circular(5)), color:
Colors.orangeAccent,)),
        ],),
        const SizedBox(height: 8),
        Row(
            children: [
                StreamBuilder(
                    stream: ref1.onValue,
                    builder: (BuildContext context,
AsyncSnapshot<dynamic> snapshot) {
                        if(snapshot.hasData){
                            var cal = snapshot.data.snapshot.value['calories'];
                            var calories = cal+300;
                            double cards = ((calories*0.44)/4);
                            return Text("\${cards.round()}g | 44%", );
                        }return const Text("Loading");},
                    ),),],
            const SizedBox(width: 8),
            Column(children: [
                Text("Fat",style: GoogleFonts.abel(fontWeight:
FontWeight.bold, fontSize: 17, color: Colors.black)),
                const SizedBox(height: 2),

```

```

        Row(
          children: [
            Stack(children: <Widget>[
              Container(height: 10, width: width/5, decoration: const
BoxDecoration(borderRadius: BorderRadius.all(Radius.circular(5)), color:
Colors.black12,)),
              Container(height: 10, width: width/11, decoration: const
BoxDecoration(borderRadius: BorderRadius.all(Radius.circular(5)), color:
Colors.purple,)),
            ],),
          ],
        ),
        const SizedBox(height: 8),
        Row(
          children: [
            StreamBuilder(
              stream: ref1.onValue,
              builder: (BuildContext context,
AsyncSnapshot<dynamic> snapshot) {
                if(snapshot.hasData){
                  var cal = snapshot.data.snapshot.value['calories'];
                  var calories = cal+300;
                  double fat = (calories*0.30)/4;
                  return Text("\${fat.round()}g | 30%", );
                }
                return const Text("Loading");
              },),)],
        const SizedBox(width: 8),
        Column(children: [
          Text("Protein", style: GoogleFonts.abel(fontWeight:
FontWeight.bold, fontSize: 17, color: Colors.black)),
          const SizedBox(height: 2),
          Row(
            children: [

```



```

        Text("Note: As you are in 2nd Trimester \n300 extra calories
required",
        style: GoogleFonts.catamaran( fontSize: 15 ,fontWeight:
FontWeight.w600)),
        ],),)],),),); }
    else if (specondition == "27-40 weeks"){
    return Padding(
        padding: const EdgeInsets.only(left: 8.0,right: 8),
        child: Container(width: width/2, height: width/1.7,
        decoration: const BoxDecoration(color: Colors.white, shape:
BoxShape.rectangle, borderRadius: BorderRadius.all(Radius.circular(10))),),
        child: Column(crossAxisAlignment: CrossAxisAlignment.start,
        children: [
            Padding(padding: const EdgeInsets.only(left: 10,top: 6), child:
Text("Macros Ratio",style: GoogleFonts.abel(fontSize: 20, color:
Colors.grey.shade800, fontWeight: FontWeight.bold))),),
            Padding(padding: const EdgeInsets.only(top: 20,right: 10,left:
10),
                child: Column(
                    children: [
                        Row(
                            children: [
                                const SizedBox(width: 15,),
                                Text("Calories",style: GoogleFonts.abel(fontWeight:
FontWeight.bold, fontSize: 17, color: Colors.black),),
                                SizedBox(width: width/1.85,),
                                StreamBuilder(
                                    stream: ref1.onValue,
                                    builder: (context, AsyncSnapshot snapshot) {
                                        if (snapshot.hasData && snapshot.data != null) {
                                            var cal = snapshot.data.snapshot.value['calories'];
                                            var usercalories = cal+400;

```

```

        return Text("$usercalories Cal", style:
GoogleFonts.abel(fontWeight: FontWeight.bold, fontSize: 17, color:
Colors.grey.shade800),);}

        else{return const Text("Loading" ,style:
TextStyle(fontSize: 14.0, fontWeight: FontWeight.bold));}}

    ),
  ],
),
const SizedBox(height: 5),
Container(height: 10, width: width/1.2, decoration:
BoxDecoration(borderRadius: BorderRadius.circular(20), color: Colors.green,)),
],
),
),
const SizedBox(height: 15),
GestureDetector(
  onTap: (){
    Navigator.push(context, MaterialPageRoute(builder:
(context)=> const Health_blog()));
  },
  child: Padding(
    padding: const EdgeInsets.only(left: 30,right: 20),
    child: Row(
      mainAxisAlignment: MainAxisAlignment.spaceBetween,
      children: [

        Column(children: [
          Text("Cards",style: GoogleFonts.abel(fontWeight:
FontWeight.bold, fontSize: 17, color: Colors.black)),
          const SizedBox(height: 2),
          Stack(children: <Widget>[
            Container(height: 10, width: width/5,decoration: const
BoxDecoration(borderRadius: BorderRadius.all(Radius.circular(5)), color:
Colors.black12,)),

```

```

        Container(height: 10, width: width/8, decoration: const
BoxDecoration(borderRadius: BorderRadius.all(Radius.circular(5)), color:
Colors.orangeAccent,)),
    ],),
    const SizedBox(height: 8),
    Row(
      children: [
        StreamBuilder(
          stream: ref1.onValue,
          builder: (BuildContext context,
AsyncSnapshot<dynamic> snapshot) {
            if(snapshot.hasData){
              var cal = snapshot.data.snapshot.value['calories'];
              var calories = cal+400;
              double cards = ((calories*0.44)/4);
              return Text("${cards.round()}g | 44%", );
            }return const Text("Loading");},
          )
        ],
      ),
    ],),
    const SizedBox(width: 8,),
    Column(children: [
      Text("Fat",style: GoogleFonts.abel(fontWeight:
FontWeight.bold, fontSize: 17, color: Colors.black)),
      const SizedBox(height: 2),
      Row(
        children: [
          Stack(children: <Widget>[
            Container(height: 10, width: width/5,decoration: const
BoxDecoration(borderRadius: BorderRadius.all(Radius.circular(5)), color:
Colors.black12,)),

```

```

        Container(height: 10, width: width/11, decoration: const
BoxDecoration(borderRadius: BorderRadius.all(Radius.circular(5)), color:
Colors.purple,)),
    ],),
  ],
),
const SizedBox(height: 8),
Row(
  children: [
    StreamBuilder(
      stream: ref1.onValue,
      builder: (BuildContext context,
AsyncSnapshot<dynamic> snapshot) {
        if(snapshot.hasData){
          var cal = snapshot.data.snapshot.value['calories'];
          var calories = cal+400;
          double fat = (calories*0.30)/4;
          return Text("${fat.round()}g | 30%", );
        }
        return const Text("Loading");
      },)),),
const SizedBox(width: 8,),
Column(children: [
  Text("Protein",style: GoogleFonts.abel(fontWeight:
FontWeight.bold, fontSize: 17, color: Colors.black)),
  const SizedBox(height: 2),
  Row(
    children: [
      Stack(children: <Widget>[
        Container(height: 10, width: width/5,decoration: const
BoxDecoration(borderRadius: BorderRadius.all(Radius.circular(5)), color:
Colors.black12,)),

```



```

        ],),) ],),),); }}}
    return Center(child: Text("Loading" , style: headingStyle,));
  }),
  const SizedBox(height: 10,),
  Diet_user()],),);}

```

```

class Diet_user extends StatelessWidget {...}

```

```

Bclass calories_user extends StatelessWidget {...}

```

```

class none_conditon extends StatefulWidget {...}

```

```

Bclass _none_conditonState extends State<none_conditon> {...}

```


```

Eclass bmi extends StatelessWidget {...}
}

```

B. SCREENSHOTS

4:06 PM | 0.1KB/s



Welcome!

Sign in to continue

@ Email


Enter email e.g XX@gmail.com


🔒 Password

Enter Password with minimum 5 characters


Continue

Don't have an account [Sign Up](#)

 Sign in with Google

 Sign in with Apple

4:06 PM | 0.4KB/s



Welcome!

Sign up for continue

@ Full Name

Enter Full name

@ Email


Enter email e.g XX@gmail.com

🔒 Password

Enter Password with mininum 5 characters

Sign Up

Have an account [Login?](#)

 Sign in with Google


 Sign in with Apple

Fig no 8.1 screenshot of login page and sign up page

4:07 PM | 0.0KB/s

Details

Age 21 2/2

Height 170 3/3

Weight 65 2/2

Activity: Moderately

Gender: Female

Any Medical Condition we should be aware of:

PCOD

Specific condition

None

Goal in your mind

Maintain Weight

Fig no 8.2 screenshot of input details of user

4:07 PM | 2.9KB/s

Diet

APR 12 WED APR 13 THU APR 14 FRI APR 15 SAT APR 16 SUN APR 17 MON

Good evening
Hi Kalyan Kumar 🙌

Calories
Remaining = Goal - Food + Exercise

Calories 2041

CARDS 332g left
FATS 128g left
PROTEIN 77g left

Balanced Diet Edit Goal

Tracker Now

Burn At Least 542 Cal

Track Your steps Connect Now

Home

4:07 PM | 0.0KB/s

Diet

Drink 13 glasses of water

View All Tracker

Recipes for You

BREAKFAST
Palak Paneer
250-300 kcal
10 min

Lunch
Pesto Pasta
612 kcal
15 min

DIN
Pe
612

Today's Work Out

Home

Fig no 8.3 screenshot of Homepage of application

C. RESEARCH PAPER

Submission date: 29-Mar-2023 04:54PM (UTC+1100)

Submission ID: 2049762904

File name: Research_paper_for_major_project.docx (452.19K)

Word count: 2956

Character count: 15703

A DIET COUNSELOR ANDROID APPLICATION

Bedugam Kalyan Kumar
Dept. of CSE,
Sathyabama Institute of Science
and Technology,
Chennai, India,
Email-id:
kalyan91999@gmail.com

P Venkata Narendar
Dept. of CSE,
Sathyabama Institute of Science
and Technology,
Chennai, India
Email-id:
narendarpsr@gmail.com

Dr. Jemshia miriam
Associate professor,
Dept. of CSE, Sathyabama
Institute of Science and
Technology, Chennai, India
Email-id:
jemshia.cse@sathyabama.ac.in

Abstract— The modern trend is to lead a more physically active and healthier existence. The style of life in practically every home has altered as a result of this development. Thus, in order to live a well-rounded & life opportunities in such a busy and demanding atmosphere, healthier dietary & nutritional cuisine have become crucial components of everyone's daily routine. However, an unhealthy diet may have serious consequences for one's physical and mental health, increasing the risk of developing conditions like obesity, diabetes, and other metabolic disorders. Normal illnesses and their symptoms may be mitigated or avoided entirely via regular exercise and a more healthful diet. And thus, those who want to eat well have turned to the web to learn about the nutritional content of various foods. Organs and tissues can only function at peak efficiency when they are supplied with the nutrients they need from the food we eat. Calories are essential to our development and vitality. Through careful dietary management, we may control our caloric intake to meet our needs. This software offers a sophisticated algorithm that can tailor a healthy eating plan to the user's specific stats (length, bodyweight, BMI, sexuality, etc.). It also connects the user to a real-life nutritionist for advice if they have food allergies. There is also a page for casual reading on healthcare as well as the physical figure.

Keywords— BMI, BMR, Sexuality, Age, weight, height, nutrition, dietitian.

INTRODUCTION

The android app NutriDiet helps its customers create individualised diet plans. It functions as a nutrition consultant 'ssimilar to a genuine Dietitian. This system functions similarly to that of a professional nutritionist. The dietitian has to know the individual's height, gender, ethnicity, and other characteristics in order to create a customized eating plan. In a similar vein, this technology generates a diet plan based on the user's input. The user is presented with a diet plan once the system has collected and processed all relevant information from them. In order to use the mission's features, users must first create an account using the login page. That way, the customer doesn't have to waste time going to see a dietitian; instead, they may quickly and easily get the diet plan they need with just a few clicks.

The primary objective of this initiative is to help individuals who have trouble getting to their doctor regularly to inquire about their nutrition receive the care they need. With this in mind, it's clear that the use of new gadgets and mobile apps, as well as their development in entrepreneurship, plays a significant role in ensuring that people have access to a healthy diet and sufficient information about the features and benefits of a variety of different health and nutrition apps with just the click of a button. Diet planner, food journal, calorie counter, weight reduction plan, and a special diet for expectant mothers are just few of the features included in this app. Furthermore, familiarity with fundamentals of human nutrition is required. Personal nourishment is a branch of medicine concerned with the effects of food and drink on physical health and fitness. Humans and plants alike rely on nutrition or nutrients to thrive. In nutrition, the term "suitable cuisine" often refers to one that is high in both micronutrients and macronutrients. Flurry Analytics reports that "the mobile development industry experienced an abrupt 62% growth in the consumption of health applications in 2016". Several medical apps in the Android marketplace present their consumers with meal plans. Unfortunately, there is currently no software that serves as both a diet planner and an activity

monitor. In light of this, we have developed an Android app called NutriDiet to aid users in their pursuit of a better lifestyle via a well-balanced diet and general fitness. BMR, or basic energy expenditure, is the amount of energy our body uses just to stay alive. One of the early formulae used to determine BMR was the Harris-Benedict's Formula.

$$\text{BMR} = 66.47 + (13.75 \times \text{weight in kg}) + (5.00 \times \text{height in cm}) - (6.75 \times \text{age in years}) \quad \text{Men} \\ \text{BMR} = 645.6 + (9.24 \times \text{weight in kg}) + (3.09 \times \text{height in cm}) - (4.77 \times \text{age in years}) \quad \text{Women}$$

After determining the user's daily caloric intake, the method identifies him or her as underweight, normal, or overweight depending on his or her body mass index. The software will provide the user a diet plan dependent on the category they choose. Qualified nourishment experts & physicians analyzed & designed the diet recommendations used in the plan. Users are able to sign up for the Personalised Nutritionist service once we make it available to them. Hypoglycemic, hypertensive, and women with irregular menstrual cycles need a special cuisine. Keep a calorie log of your eating & exercising habits.

More precise results may be expected since the system takes the user's input as given, evaluates it in light of certain metrics previously known to the programme, and then presents the user with a proposed diet plan for approval. A different diet plan may be suggested by the system if the first one is rejected. An individual who is interested in maintaining a healthy weight and lifestyle may do so by adhering to the plan that is made available to him. The Health Facts card, accessible from the main screen of the app, contains fascinating information on the human body and its components and is a great resource for learning about health in general. If the user wants to keep his body in tip-top shape, then he should utilise this software as well as adhere to the food and exercise routine it suggests.

I. LITERATURE REVIEW

In the survey phase, we identified the development's overarching objective & began the hunt for relevant scholarly articles that would aid us in developing the software. We looked through a wide variety of IEEE and Bayes articles and came across several that were relevant to our health-related research topic.[1,2,3] We obtained information from a wide variety of articles, some of which were more complex than others but all of which were intriguing in their own ways. The current health care important primary necessity and drawback is that every consultation must take place in person between the physician & the patient. To obtain help with your nutrition under the existing arrangement, you have to pay a dietician. The data may be misinterpreted, and mistakes are likely to be made. It's also a lengthy process. Conventional methods of governance have become obsolete due to the ever-increasing patient load in healthcare facilities. Since this is the case, there has been a growing need for a cutting-edge healthcare administration system.

Certain initiatives were web-based while others relied on mobile apps; some were created specifically for the goal of treating a specific ailment, such as overweight, grave

abnormalities, etc.[4] Our goal in using Android for this project was to provide a user-friendly and intuitive interface. Seeing as how there were both free and paid options, we aimed to make ours the latter. We began researching the existing model & how it functions, as well as how a genuine dietician does their job and determines a person's diet based on their specifics (length, bodyweight, maturity level, sex, etc.).[5,6] A lot of the calculations we used to figure out our daily caloric intake and overall intake were found on the internet. What one eats relies entirely on how active one is on an average day.

The number of calories burned depends on the individual's activity level, so it's important to take it into account when determining how many calories someone burns in a day[7,8,9]. The ideal ratio of amino, glucose, and fats out of the total number of calories ingested is 2:1:1. [10]The current health care system's most important necessity and drawback is that every appointment must take place in person between the patient & the practitioner. A lot of mistakes and incorrect interpretations of data are possible, too. It's also slow and annoyingly complicated.

[As the number of people requiring medical attention at hospitals and clinics continues to rise, the old ways of handling things have been ineffective[11]. Since this is the case, there has been a growing need for a cutting-edge healthcare administration system.

II. INFERENCES OF EXISTING SYSTEM

The current system is built on existing algorithms that are content-based. Food recipes are recommended using a content-based approach that takes into account the information about the user. The user's chosen recipes are divided up into ingredients, each of which is given a grade based on the interests of previously saved users. Recipes using the appropriate ingredient are suggested. Conventional approaches do not take into account dietary balance and nutrition-related aspects. Nutritional aspects that are crucial for recommending foods and a balanced diet are overlooked.

One can only search for milk- or fish-containing foods, only shows the food's nutritional content, only has the most well-liked foods, not a vast range. A tiny dataset is used for the current system of customised diet advice system, and just one or two characteristics, such as weight reduction, are taken into account. There are several healthcare apps accessible that provide consumers diet programmes. However, there is currently no such programme that can provide its user both an activity tracker and a diet plan. Consequently, to improve their encounter and make their workout regime a little easier. Whereas Fitness pal and the Balanced Diet app only offer calorie information, they also include meal planners and other tools for weight loss.

III. DESIGN & IMPLEMENTATION

We addressed the big concept of the idea and how to put it into action in this. The modules that have been explored for

eventual implementation are mentioned along with some background information.

- 1) Outline
- 2) Image
- 3) Objectives
- 4) Footfall count

First, during the Design Phase, we sketched down the sequence of operations that would occur when using the application, as shown in the diagram below. As the first phase of the Planning Phase, we planned out the sequence of operations that would occur within the application, as depicted in the figure underneath. Since we need user registration in order to gather data for calculating statistics, we built our signup and login process utilizing Google's Firebase - an open-source cloud-based infrastructure. One that uses the cloud as a repository for our credentials. It has a simple interface, and the documentation is helpful. The app will collect personal information from users throughout the enrollment stage and save it to a firebase database. After logging in, the user will be able to see his information on a panel card and make any necessary changes. User registration information will be used to populate the Diet Plan card with the user's specific information. The results of all the computations were obtained after extensive investigation and the development of appropriate formulae. The recycle aspect of Android's architecture project was used to create the Fitness card, which displays a list of objects that can be edited and revised as needed.



Fig.1- Flow Diagram

IV. ENHANCEMENTS TO EXISTING SYSTEM

- A. Integration with movable electronic devices and intelligent scales.
- B. Customization using programmes that use ml techniques,

C. Meal planning and recipe suggestions built right into the app.

D. Incorporation of dining establishments and delivery services for food.

E. Keeping tabs on the amounts of macro- and micronutrients consumed.

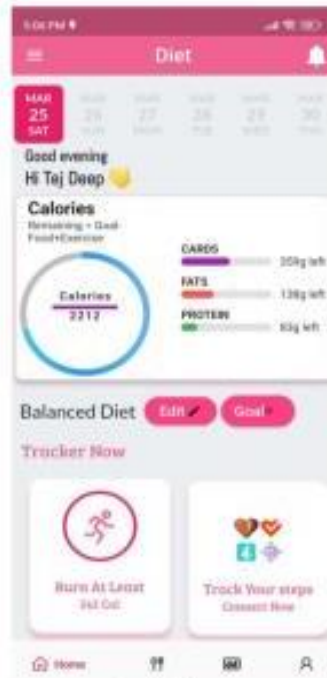
F. Social features for support and accountability.

The implementation - Carried out using the assistance of the referenced documents that were employed as well as the web, and Google's own exclusive App is used to construct the software. We were able to incorporate some design aspects into our app with the assistance of the Studio Docs. In addition, we were able to add Interfaces for the sql databases & authorization. We were able to construct a reliable application by following the instructions provided in a few different tutorials. After doing some investigation on the online research, we were able to get the formulae that we needed in order to compute the BMI and the total number of calories that we should consume.

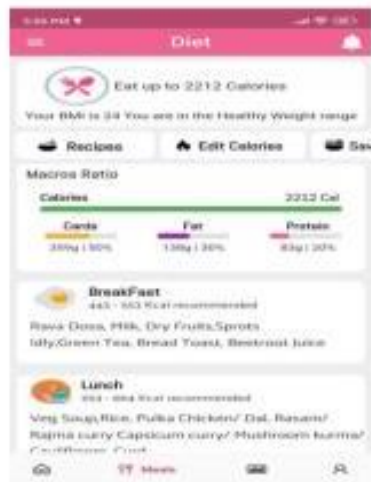
Determining Diet and Offering Diet : Customers' Upkeep calories need to be determined based on their stature, height, gender, and other factors. The diet itself has to be provided based on those calculations. The formula shown above is used to compute the Maintenance Calories. After the calories have been computed, the system examines the user's BMI to determine whether or not the user falls into the underweight, normal, or overweight categories. The user is then provided recommendations for the sort of diet programme that he or she should begin. However, the person is still offered the option to choose a diet category such as Lose more weight, Maintain Gravity, or Lose More weight. The user is given a diet to follow by the software, and it is determined by the category they choose. The Diet is founded on the records of foods that have been computed and prepared by knowledgeable Nutritionists & Dietitian located all over the world. The Customer continues to have the choice of speaking with a tangible health professional if he or she has some problems with the diet that is being provided or has specific food intolerances or irritations. For instance, some people have lactose intolerance, so the nutritionist will propose some successor for the milk protein products that are included in the diet.

RESULTS & DISCUSSIONS

Snapshots of UI for the system as follows:



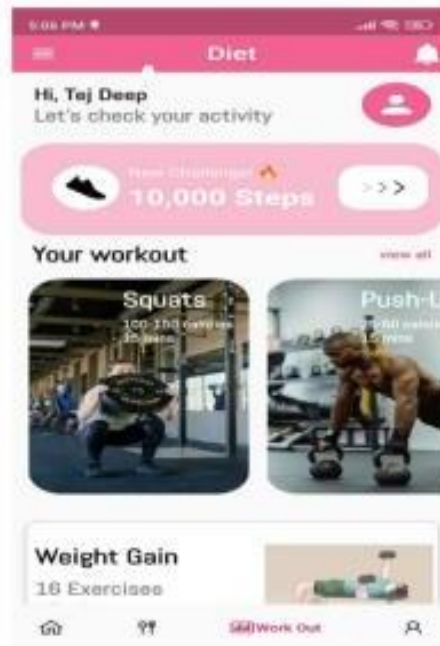
Total calories



Calories to burn



Diet menu



Workout page

Basic Information

Full name: Taj deep

Gender: **Male** Female

Email: taj@gmail.com

Age in years: 25

Daily activity: Moderately

Weight: 185 cms

Current Weight: 65 kgs

Age: 24

Medical Conditions: None

Any medical issues we should know about?

Profile

V. CONCLUSION

We would be able to place the NutriDiet App in the center of every person's hand if we are successful in developing the application as planned. Through the program's future revisions, it will be possible to put the program on the web by integrating its many servers onto the cloud. With regard to the comments and suggestions made by app users, more enhancements might be implemented into the software in order to make it so much more user pleasant. As part of our strategy for putting this initiative into action, we have developed an android app that serves as a virtual nutritionist. The primary components of our system are a customer logon as well as an administrator login, among other things. The user is able to establish their own profiles inside the software suite, as well as submit all of their statistics, and the system will compute their BMI. The administrator has the ability to inspect each user's profile and delete any accounts that are found to be inaccurate. People who have a major health care need but are also very occupied with their calendars are able to begin utilising our application and begin observing the nutrition and exercise regimens even though they are quite busy. The client of this programme doesn't need to go to a real dietician since he/she will have a nutritionist right there in the palms of his hands owing to this helpful tool.

VI. FUTURE WORK

The customer group's input, in the form of comments and recommendations, may help this software become even better. This software has the potential to be made better with

the assistance of a qualified nutritionist who is able to assist us in developing a variety of programmes that are suited to certain user categories. The scope of the project may be readily expanded, and its quality can be raised by the implementation of further iterative versions of the project. We want to concentrate our efforts on enhancing the network efficiency in its entirety. In addition to this, there will be an emphasis placed on contact here among companion and the nutritionist through teleconferencing via encrypted prescriptions. The emphasis will shift to include several more routes to become a dietician.

REFERENCES

- [1] Prof. DV Chandran, Sayali Adarkar, Apoorva Joshi, Preeti Kajbajda, "Digital Medicine: An android based application for health care system", IRJET, Volume-4, 04Apr-2017.
- [2]Hilde A.E. Geraedts, Wiebren Zijlstra, Wei Zhang, Sophie L.W. Spoorenberg, "Home Based Exercise Program Driven By Tablet Application & Mobility Monitoring, Public Health Research", Volume 14-E12, Feb-2017.
- [3]Rodrigo Zemin Franco, Julie Anne Lovegrove, Rosalind Fallaize, Foustina Hwang, Popular Nutrition-Related Mobile Apps: A Feature Assessment, JMIR MHEALTH & UHEALTH, Volume-4 Issue-3, Aug-2016.
- [4] Oxford handbook of nutrition and dietetics edited by Joan Webster-Gandy, Angela Madden and Michelle Holdsworth
- [5] Hong, S. M. and Kim, G. (2005). 'Web Expert System for Nutrition Counseling and Menu Management,' J Community Nutrition, 72, 107-113.
- [6] Hom, W., C. Popow, S. Miksch, and A. Seyfang. (2002), 'Benefits of a Knowledge-based System for Parenteral Nutrition Support: a Report after 5 Years of Rostin Daily Use,' Proceedings of the 15th European Conference on Artificial Intelligence (ECAI 2002), van Hamelen, F. (Ed.), 613-617.
- [7] Khan, A. and Hoffmann, A. (2003). 'Building a case-based diet recommendation system without a knowledge engineer,' Artificial Intelligence in Medicine, 27, 155-179.
- [8] Chen, Y. , Hsu, C. Y. , Liu, . L and Yang, S. (2012), 'Constructing a nutrition diagnosis expert system,' Expert System with Application, 39(2) , 2132-2156
- [9] Becerra-Fernandez, I., Gonzalez, A., & Sabherwal, R. (2004). Knowledge management: Challenges, solutions and technologies. New Jersey: Pearson Education Inc.
- [10] Balintfy JL. Menu planning by computer. Communications of the ACM. 1964;7(4):255-259. doi: 10.1145/364005.364087.
- [11] Heart, Lung, and Blood Institute; National Institutes of Health, authors. Interactive menu planner.

A DIET COUNSELOR ANDROID APPLICATION

ORIGINALITY REPORT

3%

SIMILARITY INDEX

3%

INTERNET SOURCES

2%

PUBLICATIONS

2%

STUDENT PAPERS

PRIMARY SOURCES

1

ijeast.com

Internet Source

1%

2

M. Jalsari, L. Lakshmanan. "A Survey: Integration of IoT and Fog Computing", 2018 Second International Conference on Green Computing and Internet of Things (ICGCIoT), 2018

Publication

1%

3

Samireddy Adithya, Tudimaladinna Sanjay Rezinald, A. Viji Amutha Mary, J. Refonaa, S. L. Jany Shabu, P. Jeyanthi. "Child Mortality Prediction using Machine Learning", 2022 6th International Conference on Trends in Electronics and Informatics (ICOEI), 2022

Publication

<1%

Exclude quotes On

Exclude matches Off

Exclude bibliography On