

NutriLife: Empowering Health with Diet and Nutrition App

By

Yuki Tan Lok Yee

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONOURS)

Faculty of Information and Communication Technology

(Kampar Campus)

JUNE 2024

REPORT STATUS DECLARATION FORM

Title: NutriLife: Empowering Health with Diet and Nutrition App

Academic Session: June 2024

I _____ YUKI TAN LOK YEE _____
(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.



(Author's signature)

Verified by,



(Supervisor's signature)

Address:

16, Jalan Pengkalan Tiara 15,
Taman Pengkalan Tiara, 31650
Ipoh, Perak

____ Ts Wong Chee Siang____
Supervisor's name

Date: 13th September 2024

Date: 13th September 2024

Universiti Tunku Abdul Rahman			
Form Title : Sample of Submission Sheet for FYP/Dissertation/Thesis			
Form Number: FM-IAD-004	Rev No.: 0	Effective Date: 21 JUNE 2011	Page No.: 1 of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

UNIVERSITI TUNKU ABDUL RAHMAN

Date: 13th September 2024

SUBMISSION OF FINAL YEAR PROJECT /DISSERTATION/THESIS

It is hereby certified that YUKI TAN LOK YEE (ID No:20ACB05030) has completed this final year project entitled “NutriLife: Empowering Health with Diet and Nutrition App” under the supervision of Ts Wong Chee Siang (Supervisor) from the Department of Computer Science, Faculty of Information and Communication Technology.

I understand that University will upload softcopy of my final year project / dissertation/ thesis* in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,



(Yuki Tan Lok Yee)

DECLARATION OF ORIGINALITY

I declare that this report entitled “**NUTRILIFE: EMPOWERING HEALTH WITH DIET AND NUTRITION APP**” is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : _____  _____

Name : Yuki Tan Lok Yee

Date : 13th September 2024

ACKNOWLEDGEMENTS

I would like to express thanks and appreciation to my supervisor, Ts Wong Chee Siang, and my moderator Ts Dr Chai Meei Tyng who have given me a golden opportunity to embark on the design of a health mobile application. The guidance, support, and mentorship have been instrumental in shaping my journey in the field of mobile application design. This project marks a significant milestone in my career, and I am deeply grateful for your trust and encouragement.

Furthermore, I am immensely thankful to my family members for their unwavering support throughout the development of this health mobile application. Their encouragement, understanding, and patience have been invaluable to me during this challenging journey. They not only provided mental and physical support but also contributed ideas that enriched the project. Their unwavering support, both emotionally and practically, has been a constant source of strength and motivation.

I am deeply grateful for their sacrifices, love, and belief in my abilities. Thank you for standing by me every step of the way and for being my pillars of support

ABSTRACT

In Malaysia, the lack of food recognition applications focused on local cuisine presents a challenge, as current solutions primarily cater to Western foods. This gap limits users' access to accurate nutritional information for Malaysian dishes, impeding their health goals and meal planning. This project addresses these issues by developing a food recognition app specifically for Malaysian cuisine. Utilizing a Convolutional Neural Network (CNN) model trained in Google Colab with a dataset of local dishes, the app offers precise nutritional insights, including calories and macronutrients, based on user-captured images. Furthermore, the app enables users to save and revisit their food records.

A major innovation of this project involves integrating computer vision and machine learning for real-time fitness guidance, specifically focusing on exercise technique. Using MediaPipe and deep learning algorithms, the app track and analyses users' movements during exercises such as squats and bicep curls, providing corrections to improve technique. This integration aims to enhance user experience and effectiveness in fitness training. Overall, this project strives to bridge gaps in food recognition and fitness tracking through advanced technologies, delivering valuable tools for health and wellness.

Table of Contents

TITLE PAGE	II
REPORT STATUS DECLARATION FORM	II
FYP THESIS SUBMISSION FORM	III
DECLARATION OF ORIGINALITY	IV
ACKNOWLEDGEMENTS	V
ABSTRACT	VI
LIST OF FIGURES	X
LIST OF TABLES	XIII
LIST OF ABBREVIATIONS	XIV
CHAPTER 1 INTRODUCTION	1
1.1 Project Background.....	1
1.1.1 Artificial Intelligence (AI) Fitness Tracking	4
1.2 Problem Statement and Motivation	5
i. Limited in Malaysian food database.....	5
ii. Real-time recognition with feedback are not provided.....	6
iii. Lack of motivation to attain the set goals.....	7
1.3 Objectives	9
i. To develop a mobile application with a food database that include a range of Malaysian local foods.....	9
ii. To develop a mobile application that provides real-time feedback to users.....	9
iii. To develop a mobile application that enable body mass index (BMI) calculation.....	9
iv. To develop a mobile application that provides food recognition.	9
v. To develop a mobile application that tracks cumulative days of meal recording or workouts.....	10
vi. To develop a mobile application that provides users with recommended recipes.	10
1.4 Project Scope and Direction.....	11
1.4.1 User component	11
1.4.2 Food Component.....	12

1.4.3 Fitness Component.....	13
1.5 Contribution	14
1.6 Report Organization.....	15
CHAPTER 2 LITERATURE REVIEW	17
2.1 System review on existing diet and nutrition applications	17
i. MyFitnessPal	17
ii. MyNetDiary	18
iii. Lifesum.....	19
iv. Cronometer	20
2.2 Limitations of the previous studies	21
i. MyFitnessPal	21
iii. Lifesum.....	22
iv. Cronometer	22
2.3 Compare and contrast of the previous study.....	23
CHAPTER 3 SYSTEM METHODOLOGY/APPROACH.....	24
3.1 Methodology and General Workflow Procedures	24
3.1.1 User stories.....	25
3.1.2 Tasks	27
3.1.2 Model Training	31
3.2 System Design	45
3.2.1 System Architecture Design	45
3.2.2. Use Case Diagram.....	47
3.2.3 Activity Diagram	50
3.3 Project Timeline.....	57
CHAPTER 4 SYSTEM DESIGN	58
CHAPTER 5 SYSTEM IMPLEMENTATION	61
5.1 Tools to use.....	61
5.1.1 Hardware Specifications	61
5.1.2 Software Specifications	62
5.1.3 System Requirements for Users.....	64
5.1.4 Software Setup (Food Recognition).....	65
5.1.5 Software Setup (Fitness tracking - Squat)	77
5.1.6 Software Setup (Fitness tracking – Bicep curl)	81
5.2 System Operation.....	85
CHAPTER 6 SYSTEM EVALUATION AND DISCUSSION	101

6.1 Model Training and Performance	101
6.1.1 Food Recognition.....	101
6.1.2 Fitness tracking	103
6.2 System performance.....	104
CHAPTER 7 CONCLUSION AND RECOMMENDATION.....	108
7.1 Conclusion	108
7.2 Recommendations.....	109
REFERENCES.....	110
APPENDIX.....	111

LIST OF FIGURES

Figure Number	Title	Page
Figure 1.1.0	Obesity rate in Malaysia(%)	1
Figure 1.1.1	Diabetes prevalence rate in Malaysia (%)	2
Figure 1.1.2	Prevalence of underweight and obesity among adults aged 18 years and older, globally by WHO region, 1990-2022	2
Figure 1.1.3	of thinness and overweight among children and adolescents aged 5-19 years, globally by WHO region, 1990-2022	3
Figure 3.3.1	The extreme programming release cycle	24
Figure 3.1.2.1	Eight distinct classes of Malaysian Food	31
Figure 3.1.2.2	Example datasets collected for Nasi Lemak class	31
Figure 3.1.2.4	Confusion matrix for food recognition model of NutriLife	35
Figure 3.1.2.2.1	Calculation_angle function's equation	39
Figure 3.1.2.2.2	Pose landmarks	44
Figure 3.2.1.0	System Architecture Design	45
Figure 3.2.2.0	Use Case Diagram for NutriLife	47
Figure 3.2.3.0	Log in and sign-up activity diagram	50
Figure 3.2.3.1	Food recognition activity diagram	51
Figure 3.2.3.2	BMI activity diagram	52
Figure 3.2.3.3	Food page activity diagram	53
Figure 3.2.3.4	Fitness page activity diagram	54
Figure 3.2.3.5	Statistic page activity diagram	55
Figure 3.2.3.5	Home page activity diagram	56
Figure 3.3.0	Project Timeline for NutriLife	57
Figure 4.1.0	System flowchart for NutriLife	58
Figure 5.1.4.0	Python and TensorFlow Installation	65
Figure 5.1.4.1	Install tensorflow in google colab	65
Figure 5.1.4.2	Dataset preparation	65
Figure 5.1.4.3	Pre requisites and GPU configuration	66
Figure 5.1.4.4	Mount from google drive	66
Figure 5.1.4.5	To verify correct folder is being extracted	66
Figure 5.1.4.6	Remove dodgy images	67
Figure 5.1.4.7	Split the datasets	68
Figure 5.1.4.8	Data loading and preparation	69
Figure 5.1.4.9	Build the CNN layer	70
Figure 5.1.4.10	Convert the image into RGBA model	72
Figure 5.1.4.11	Compiling and training the model	72
Figure 5.1.4.12	Running epochs	73
Figure 5.1.4.13	Accuracy of 74% for the model	74
Figure 5.1.4.14	Illustrate the model accuracy and model loss graph	74
Figure 5.1.4.15	Graph of model accuracy and model loss	74
Figure 5.1.4.16	Plotting the confusion matrix	75
Figure 5.1.4.17	The confusion matrix	75
Figure 5.1.4.18	Save and export model in TensorFlow Lite format	76
Figure 5.1.5.0	Data collection and model training for squat	77
Figure 5.1.5.1(a)	Conversion of Trained Model to TensorFlow Lite Format	78
Figure 5.1.5.1(b)	Conversion of Trained Model to TensorFlow Lite Format	79
Figure 5.1.5.2	Save model	80
Figure 5.1.6.0	Data collection and model training for bicep curl	81

Figure 5.1.6.1 (a)	Conversion of Trained Model to TensorFlow Lite Format	82
Figure 5.1.6.1 (b)	Conversion of Trained Model to TensorFlow Lite Format	83
Figure 5.1.6.2	Save model	84
Figure 5.2.1	Splash Screen	85
Figure 5.2.2	Login Page	85
Figure 5.2.3	Sign up Page	86
Figure 5.2.4	Finished registered	86
Figure 5.2.5	Introduction of the main highlighted features in NutriLife	87
Figure 5.2.6	Introduction of the main highlighted features in NutriLife	87
Figure 5.2.7	Introduction of the main highlighted features in NutriLife	87
Figure 5.2.8	Introduction of the main highlighted features in NutriLife	87
Figure 5.2.9	Homepage	88
Figure 5.2.10	Statistic Page	88
Figure 5.2.11	Cook Page	89
Figure 5.2.12	Cook Page	89
Figure 5.2.13	Example of detailed information of recommended meal for breakfast	90
Figure 5.2.14	Example of detailed information of recommended meal for lunch	90
Figure 5.2.15	Example of detailed information of recommended meal for dinner	90
Figure 5.2.16 (a)	Food recognition page	91
Figure 5.2.16 (b)	Food recognition page	91
Figure 5.2.17	Take photo options	92
Figure 5.2.18 (a)	Upload from gallery options	92
Figure 5.2.18 (b)	Upload from gallery options	93
Figure 5.2.19	Calories and macronutrients of the food uploaded	93
Figure 5.2.20	Food not recognized	93
Figure 5.2.21	Food info page	93
Figure 5.2.22	Edit the food's information	94
Figure 5.2.23	The information of the food is being uploaded	94
Figure 5.2.24	Predicted food based on the food taken or uploaded, values of the calories and macronutrients are set to zero by default	95
Figure 5.2.25	Predicted food based on the food taken or uploaded, values of the calories and macronutrients are set to the average estimated values	95
Figure 5.2.27 (a)	Example of food records saved on 1st September 2024	96
Figure 5.2.27 (b)	Example of food records saved on 1st September 2024	96
Figure 5.2.28 (a)	Example of food records saved on 11th September 2024	96
Figure 5.2.28 (b)	Example of food records saved on 11th September 2024	96
Figure 5.2.29	Fitness Page	97
Figure 5.2.30	Squat	97
Figure 5.2.31	Bicep curl	97
Figure 5.2.32 (a)	Bicep curl	98
Figure 5.2.32 (b)	Bicep curl	98
Figure 5.2.33(a)	Squat	98
Figure 5.2.33(b)	Squat	98
Figure 5.2.34	BMI page	99
Figure 5.2.35	BMI - underweight	99

Figure 5.2.36	BMI - normal weight	99
Figure 5.2.37	BMI - overweight	99
Figure 6.1.1.0	The test accuracy of food recognition model (0.74)	101
Figure 6.1.1.1	Graph of model accuracy and model loss	101
Figure 6.1.1.2	The confusion matrix	102
Figure 6.1.2.1.0	Test accuracy of Squat model (0.9969)	103
Figure 6.1.2.2.0	Test accuracy of Bicep Curl model (0.9988)	103

LIST OF TABLES

Table Number	Title	Page
Table 2.3.1	Table of comparison between MyFitnessPal, MyNetDiary, Lifesum, and Cronometer	23
Table 3.1.2.3	Different batch size and number of datasets affecting the model's performance	33-34
Table 3.1.2.2.0	Squat data collection	36-39
Table 3.1.2.2.1	Bicep curl data collection	40-43
Table 5.1.1	Specifications of laptop	61
Table 5.1.2	Specifications of android device (Oppo Reno 7 5G)	61

LIST OF ABBREVIATIONS

<i>XP</i>	Extreme Programming
<i>PXP</i>	Personal Extreme Programming
<i>CNN</i>	Convolutional Neuro Network
<i>RGBA</i>	Red, Green, Blue, Alpha

CHAPTER 1 INTRODUCTION

1.1 Project Background

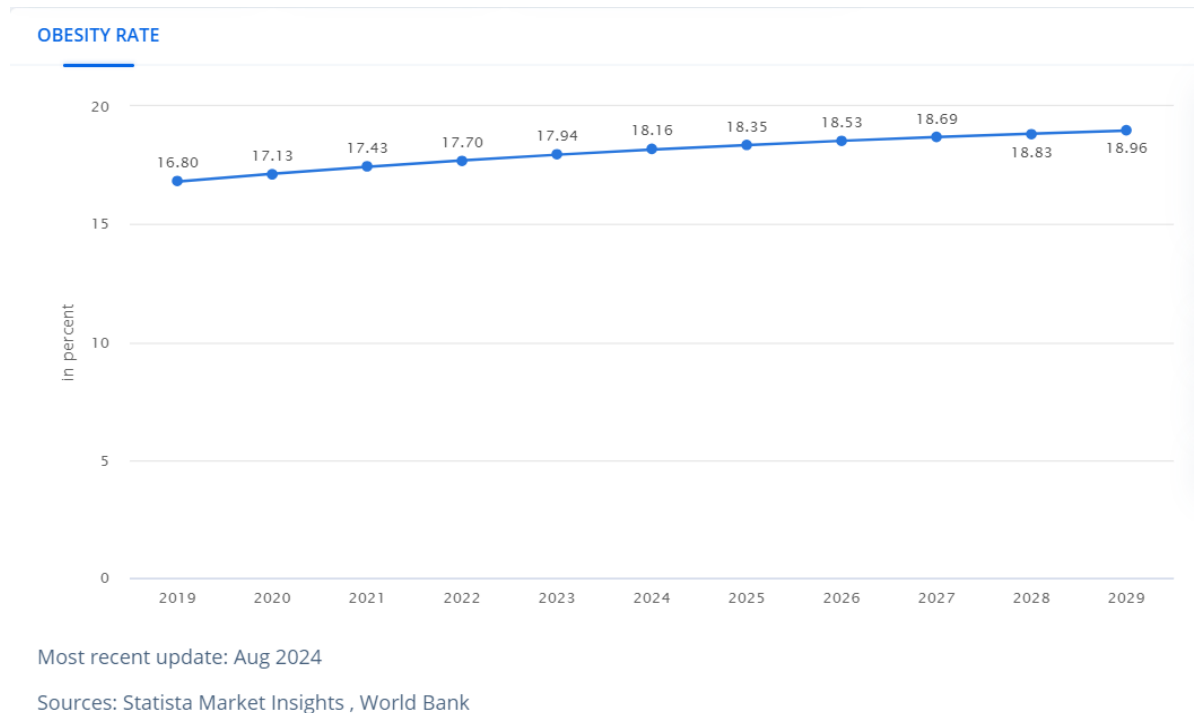


Figure 1.1.0 Obesity rate in Malaysia (%)

According to Statista Market Insights[1], the obesity rate in Malaysia has experienced a significant upward trend in recent years, as illustrated in Figure 1.1.0. Between 2019 and 2020, the obesity rate increased by 0.33%, rising from 16.80% to 17.13%. However, the rate of increase began to slow in the subsequent years, with a rise of 0.10% from 2020 to 2021, bringing the obesity rate to 17.43%. This trend of deceleration continued into 2022, with an increase of 0.27%, reaching 17.70%, and a further rise of 0.24% in 2023, bringing the rate to 17.94%. The projected rate for 2024 is 18.16%, with an expected increase of 0.22%.

Although the obesity rate is still on the rise, the decreasing rate of increase each year could suggest positive developments. The slowdown may be attributed to several factors, including enhanced health awareness, lifestyle changes prompted by the COVID-19 pandemic, and public health initiatives promoting healthier living. These factors may have contributed to more effective management of obesity, despite the overall rate remaining high. This data suggests that, while obesity remains a challenge,

CHAPTER 1

ongoing efforts to improve health behaviours and interventions are beginning to show positive results.

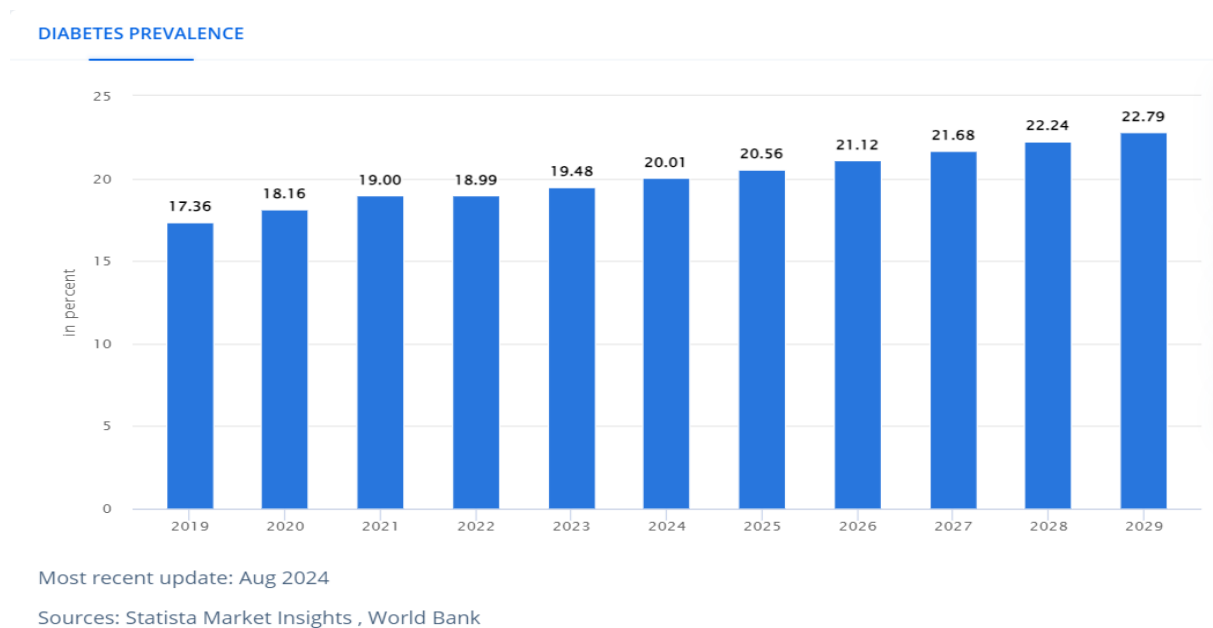


Figure 1.1.1 Diabetes prevalence rate in Malaysia (%)

In addition to the obesity issue, the prevalence of diabetes in Malaysia has also raised growing concern. According to Statista Market Insights[1], diabetes rates have steadily increased from 2019 to 2024. In 2019, the prevalence of diabetes was 17.36%, rising to 18.16% in 2020 and further to 19.00% in 2021. Although a slight decrease to 18.99% was recorded in 2022, the rate rose again to 19.48% in 2023, with a projected prevalence of 20.01% in 2024, as shown in Figure 1.1.1.

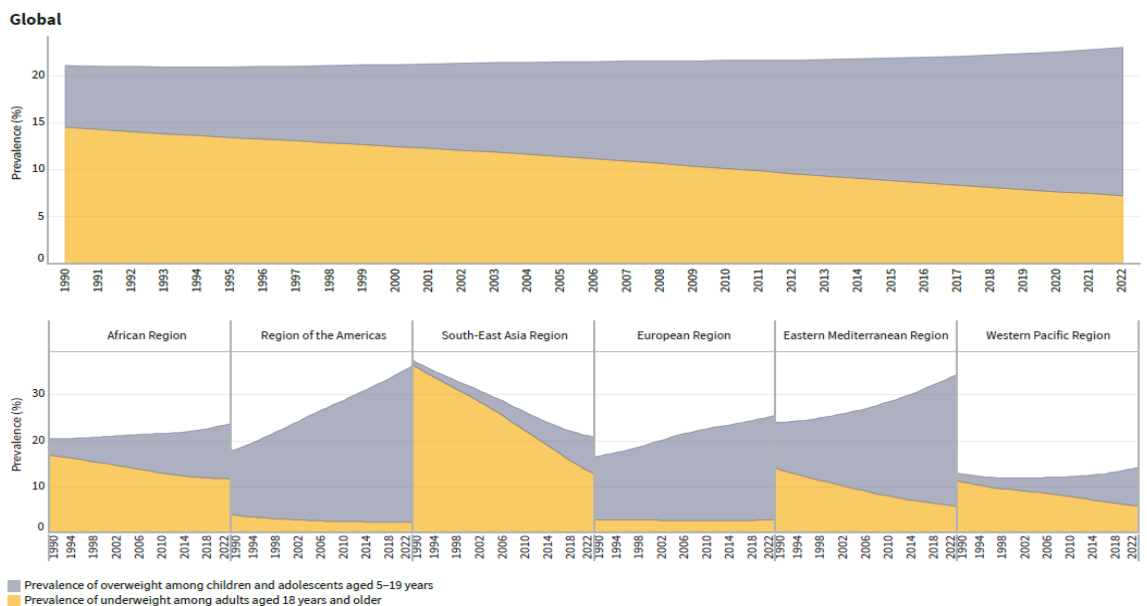


Figure 1.1.2 Prevalence of underweight and obesity among adults aged 18 years and older, globally by WHO region, 1990-2022

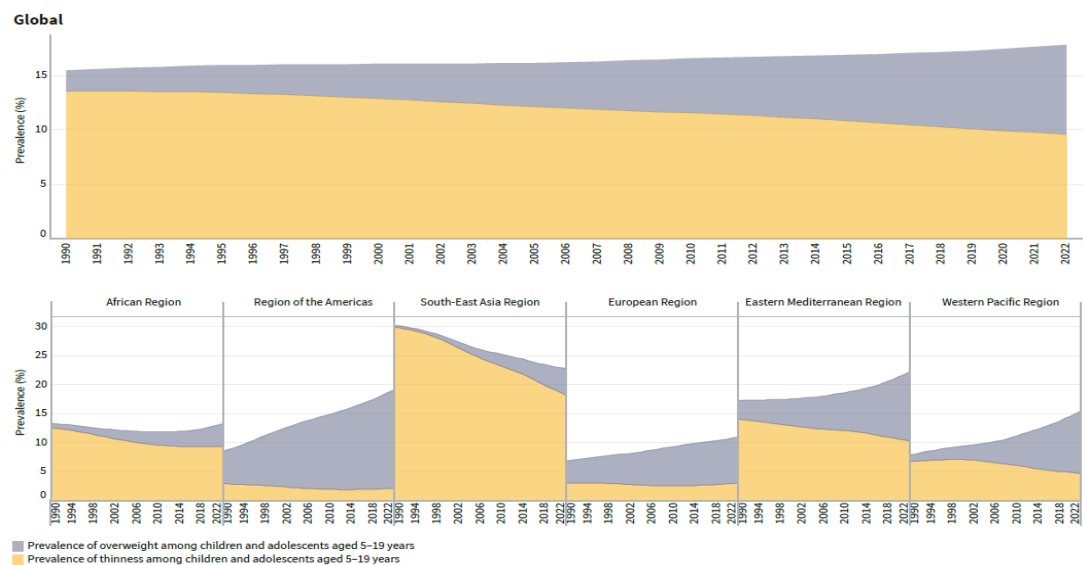


Figure 1.1.3 Prevalence of thinness and overweight among children and adolescents aged 5-19 years, globally by WHO region, 1990-2022

Additionally, according to the World Health Organization (WHO), poor nutritional habits are a major risk factor for health globally, with malnutrition taking various forms, including undernutrition, overweight (including obesity) shown in Figure 1.1.2, and micronutrient deficiencies. Between 1990 and 2022, the number of adults aged 18 years and older living with obesity rose dramatically by nearly 700 million, reaching 890 million, while those classified as underweight decreased by 90 million to 400 million as shown in Figure 1.1.3.[2]

The persistent rise in diabetes prevalence highlights a major public health challenge. This increase is closely tied to rising obesity levels, as obesity is a primary risk factor for type 2 diabetes. The upward trends in both obesity and diabetes indicate an urgent need for comprehensive and effective health interventions.

And, in the aftermath of the pandemic, individuals have become increasingly conscious of their dietary choices, with a growing recognition of the connection between nutrition and overall health. As a result, people are now seeking reliable information on their nutrient intake and the impact of food on their well-being. This

heightened awareness has created an opportunity for innovative tools like NutriLife, a diet and nutrition app designed to offer personalized, comprehensive guidance to navigate the complexities of nutrition. NutriLife helps users monitor their diet, set health goals, and track their progress, which is particularly useful for those managing weight, obesity, and diabetes.

The pandemic has not only raised awareness about the importance of nutrition but also highlighted the vulnerability of individuals with chronic illnesses, prompting many to make significant lifestyle changes. Mobile apps have become highly popular in recent years due to their ability to streamline everyday activities. Health and wellness apps, particularly those dedicated to diet and nutrition, have gained substantial recognition as users seek ways to improve their health. NutriLife aims to make a meaningful contribution to this domain by providing users with tools and resources to lead healthier lives.

Hence, NutriLife focuses on the ability to support users in managing their diets and cultivating healthier habits. Through features such as food recognition feature that estimates calories, helping users better understand their nutritional intake. Additionally, NutriLife suggests simple and easy-to-cook healthy meals, addressing the common challenge that many users face regarding a lack of knowledge about healthy eating. To further enhance users' well-being, the app incorporates AI fitness tracking, which ensures that users maintain proper posture during workouts, leading to more effective and efficient exercise routines. Despite these features, challenges such as the inconvenience of logging food entries, confusion about diet strategies, and users forgetting to record meals still persist.

1.1.1 Artificial Intelligence (AI) Fitness Tracking

In the development of AI fitness tracking systems, the process typically unfolds in several stages with the goal of optimizing workout routines, improving exercise accuracy, and reducing the risk of injury to help users achieve their fitness objectives more effectively. [3]The first stage involves data collection and model training, where vast amounts of exercise data are gathered to train pose estimation models using technologies like Mediapipe[4] and OpenCV[5]. This foundational phase ensures that

the AI system can accurately recognize and interpret various body positions and movements.

In the second stage, the trained model is integrated into a user-friendly application or platform, where Android Studio platform is used in this project. Here, the AI system provides real-time feedback on exercise form by analysing live video input from a user's camera. This feedback is crucial for users to adjust their posture and technique, promoting more effective and safe workouts.[3]

1.2 Problem Statement and Motivation

i. Limited in Malaysian food database.

The majority of nutrition applications with food recognition features predominantly include databases focused on Western cuisine, leaving a significant gap in the availability of Malaysian food databases. This limitation is evident from the literature review, which reveals that existing databases are often insufficient for users seeking to track and manage Malaysian dishes. Users frequently encounter challenges in accurately identifying and recognizing foods due to the overwhelming number of options available[6], as well as difficulties in locating specific items not included in the database[7], [8]. Moreover, users struggle with recording portion sizes and adding homemade or restaurant meals, leading some to abandon meal tracking altogether, particularly when dealing with a diverse range of foods over extended periods. . In some cases, users might even give up on tracking their meals, especially if their diet involves a variety of foods over a longer time. Interestingly, certain studies, like the one by Lieffers et al[6], [9]note that participants generally prefer larger food databases because they save time and offer convenience.

To address these issues, the motivation of this project is to develop a comprehensive food database tailored specifically to Malaysian cuisine. This novel solution aims to overcome the limitations of existing databases by utilizing Convolutional Neural Networks (CNNs) trained on Malaysian food items to streamline the food identification process. The app will feature an intuitive "Take Photo" function, allowing users to easily scan and identify Malaysian dishes with their device's camera, thus eliminating

the need for manual searches. Additionally, an "Upload from gallery" feature will enable users to contribute images and information to expand the database, enhancing its accuracy and variety.

By leveraging advanced CNN technologies and focusing on local cuisine, this approach will enable users to accurately identify and log their meals, facilitating informed dietary choices. This solution addresses the shortcomings of current food databases and aims to improve user engagement and satisfaction by offering a tailored, comprehensive tool that reflects the rich diversity of Malaysian culinary culture.

ii. Real-time recognition with feedback are not provided.

Existing fitness applications in the market primarily aim to help users manage body weight and build muscle. However, these applications often lack real-time feedback, which is crucial for users with limited fitness knowledge. The absence of real-time assessments can lead to improper exercise techniques and increase the risk of injury, such as incorrect joint angles during workouts.

Moreover, without intelligent assessment through computer vision and machine learning, these apps are unable to analyse a user's posture effectively. This results in a lack of targeted feedback, as the apps cannot highlight specific landmarks or provide visual cues to distinguish between correct and incorrect key points. Consequently, users do not receive personalized instruction or corrections, which diminishes the effectiveness of their exercise sessions and hinders their progress.

To address these limitations, there is a need for an advanced fitness application that integrates AI-driven computer vision and machine learning techniques [4], [5]. By providing real-time analysis and feedback on exercise form, the app can highlight precise landmarks and offer visual cues for correct and incorrect posture. This approach aims to enhance user safety, improve exercise accuracy, and increase the overall effectiveness of fitness practice sessions.

To address these limitations, this project proposes the development of an advanced fitness application that integrates AI-driven computer vision and machine learning

techniques. By providing real-time analysis and feedback on exercise form, the app can highlight precise landmarks and offer visual cues for correct and incorrect posture[4]. This approach aims to enhance user safety, improve exercise accuracy, and increase the overall effectiveness of fitness practice sessions, ultimately supporting users in achieving their fitness goals more safely and efficiently.

Hence, the motivation for this project stems from the need to bridge these gaps and enhance user safety and effectiveness in fitness applications. The growing interest in fitness and health, coupled with the rise in home workouts, highlights the demand for more interactive and intelligent solutions. Users are increasingly seeking tools that can offer real-time feedback and personalized guidance, especially as fitness knowledge varies widely among individuals.[3]

iii. Lack of motivation to attain the set goals.

The diet and nutrition app acts as a tool for users to improve their eating habits and overall health. This is done by setting goals, ideally resulting in desired weight changes or better health conditions. However, if the outcomes do not match the user's expectations, their motivation to stick to a healthy diet might decrease, possibly causing them to stop using the app.

Based on a conducted study [10], the research aimed to understand user behaviour. Only 57% of the participants used health apps. Reasons for not adopting or using these apps included lack of awareness, feeling like they were not necessary, and having other tools available. Some participants already had healthy habits and did not see the need for an app. Many users also lacked knowledge about how to effectively choose and use apps.

A solution to this challenge could be the incorporation of a streak counter feature. This motivational tool would be seamlessly integrated into diet and nutrition apps with the main purpose of inspiring users to consistently stick to their dietary goals and positive routines. The streak counter would visually show the consecutive days users follow their dietary plans. This visual representation would give them a clear view of their dedication and progress. The streak counter would offer a way to feel accomplished, accountable, and positively supported, all of which greatly help maintain

CHAPTER 1

motivation and foster lasting behaviour changes. This engaging streak counter would assist users in staying focused, adopting healthier habits, and reaching their wellness goals.

1.3 Objectives

- i. To develop a mobile application with a food database that include a range of Malaysian local foods.**

To develop a mobile application that includes a comprehensive food database featuring a diverse range of Malaysian local foods. The aim of this project is to address the current market gap where existing applications predominantly focus on Western cuisine. By incorporating an extensive database of Malaysian dishes, the application will provide users with accurate and relevant food recognition and nutritional information tailored to local culinary practices. This objective seeks to enhance the user experience by offering a more inclusive and culturally relevant food tracking solution.

- ii. To develop a mobile application that provides real-time feedback to users.**

To develop a mobile application that provides real-time feedback for exercise routines by implementing the computer vision and machine learning to analyse and interpret the user posture and movements during the workouts. This feature will deliver instant corrections and recommendations based on the user's exercise performance, helping them make necessary adjustments to their form and improve the overall effectiveness of their workouts which aims to enhance user engagement and safety by providing clear, visual feedback on exercise performance by highlighting correct and incorrect postures through visual cues.

- iii. To develop a mobile application that enable body mass index (BMI) calculation.**

To develop a mobile application that allows users to enter their height and weight to compute their Body Mass Index (BMI) quickly and accurately which will categorize users as underweight, normal weight, overweight, or obese, and users can change to tailored exercise and meal plans based on their individual needs.

- iv. To develop a mobile application that provides food recognition.**

To develop a mobile application that utilizes image recognition technology to identify and classify various types of Malaysian food. This feature will analyze food images captured by the user and provide detailed information on the identified foods, including

CHAPTER 1

the average estimated calories and macronutrient content such as carbohydrates, proteins, and fats, to promote informed and healthy eating habits.

- v. To develop a mobile application that tracks cumulative days of meal recording or workouts.**

To develop a mobile application that implements a streak counter feature, allowing users to view the cumulative number of days they have been using the app for meal and workout recording. This feature aims to enhance user motivation and encourage continuous engagement to achieve their set goals.

- vi. To develop a mobile application that provides users with recommended recipes.**

To develop a mobile application that provides users with recommended recipes by suggesting various types of simple and easy-to-cook dishes randomly, based on the food database available within the application.

1.4 Project Scope and Direction

1.4.1 User component

New User Registration Module

The New User Registration module enables users to sign up for a new account, granting them access to the platform's features and services. This module features a user-friendly sign-up form that allows new users to create accounts by creating a unique username, setting a secure password, and providing a valid email address. This process ensures that only authorized individuals gain access to the system, effectively preventing fraudulent activities.

Additionally, the module includes an essential email verification process. A verification link is sent to the user, requiring activation to complete the account setup. This step ensures that the created account is linked to a valid and authenticated email address. Beyond verification, the email address also helps keep users informed about the latest updates and vital notifications related to their accounts. Once the email address is successfully verified, the registration process is complete, granting users full access to the platform's features and services.

Log In Authentication Module

The Log in Authentication module is crucial for verifying the identity of registered users. It checks the user's credentials to ensure that only authorized individuals with valid login information can access the platform. If users enter incorrect information, such as an incorrect username or password, they will be prompted to try again.

To enhance security, the module includes advanced features. For instance, if users or unauthorized individuals attempt to log in multiple times unsuccessfully, the module will temporarily lock the account to protect against potential attacks. Additionally, it automatically logs out users after a period of inactivity to prevent unauthorized access if the user forgets to log out.

The module is designed to be user-friendly, offering a simple interface for a smooth login experience. Users experiencing issues with their login credentials can use the password recovery option to regain access to their accounts, making the process

convenient. Overall, the Log in Authentication module balances security and user-friendliness, ensuring the platform remains safe and reliable for all users.

User Profile Module

The User Profile module provides users with a comprehensive and user-friendly interface to personalize their profiles according to their preferences and individual needs. Personalization is key to user engagement. This module allows individuals to input and manage their health and nutrition data, dietary preferences, and fitness goals. This functionality enables NutriLife to offer tailored recommendations and insights.

1.4.2 Food Component

Food Recognition Module

The Food Recognition module leverages advanced image recognition technology to identify and analyze food items. Users can take photo or upload images of their meals, and NutriLife will detect and analyses the food. This module provides detailed nutritional information, including calorie count and macronutrient breakdown, to help users make informed dietary choices. The system also integrates with the user's profile to track food intake and offer personalized dietary recommendations based on their health and nutrition goals. The accuracy of the recognition improves over time as the AI learns from user interactions and expands its food database.

Food Recommendation Module

The Recipe Recommendation Module is designed to enhance the user experience by providing personalized recipe suggestions. The module randomly suggests a variety of simple and easy-to-cook dishes from a comprehensive food database available within the application. By analyzing user preferences and dietary goals, the module ensures that the recommendations align with individual tastes and nutritional needs. Users can explore a diverse range of recipes, making meal planning more convenient and enjoyable. The system's ability to offer random yet relevant suggestions keep the meal options fresh and engaging, encouraging users to try new dishes while maintaining a focus on simplicity and ease of preparation.

1.4.3 Fitness Component

Artificial Intelligence (AI) Fitness Tracking Module

The AI Fitness Tracking Component focuses on analyzing user posture to enhance the effectiveness and efficiency of workouts while reducing the risk of injury. Utilizing MediaPipe and computer vision technology, the component tracks and evaluates the user's posture during exercises. By providing real-time feedback and corrections, it helps users maintain proper form and prevent common workout-related injuries. The system offers personalized recommendations to improve exercise technique and overall performance, ensuring a safer and more effective fitness experience.

1.5 Contribution

This proposed project, NutriLife, offers substantial contributions to both individual users and broader society. By providing personalized recipe recommendations and real-time fitness tracking, the app enhances users' dietary habits and workout routines, contributing to overall better health and wellness.

The User Registration Module and Log In Authentication Module ensure a secure and user-friendly experience, promoting accessibility and ease of use. This encourages more individuals to engage with the app, fostering healthier lifestyles through its tailored recommendations.

The Food Recognition Component and Recipe Recommendation Module play a pivotal role in making nutritious eating accessible. By analyzing food intake and suggesting easy-to-cook dishes, the app helps users make informed dietary choices and simplifies meal planning. This not only supports individual health goals but also promotes better dietary habits within the community.

The AI Fitness Tracking Component, utilizing MediaPipe and computer vision, offers real-time posture analysis to enhance workout effectiveness and reduce injury risks. This feature provides users with precise feedback on their exercise form, ensuring a safer and more productive fitness experience.

Overall, NutriLife is a combination of personalized nutrition and fitness features that promotes a healthier lifestyle, making it a valuable tool for users seeking to improve their well-being and for society by encouraging healthier living practices.

1.6 Report Organization

This report is organized into 7 Chapters:

Chapter 1: Introduction

Chapter 1 sets the stage for the report by providing an overview of the project. It begins with the project background, explaining the context and the underlying issues the project aims to address. This is followed by the problem statement and motivation, which outline the core problem and the driving factors behind the project. The chapter then specifies the objectives, detailing the goals and intended outcomes of the project. It also defines the scope and direction of the project, clarifying its boundaries and focus. Finally, the chapter highlights the contributions of the project, summarizing its impact on users and society.

Chapter 2: Literature Review

In Chapter 2, the report reviews existing health and nutrition applications available in the market. This literature review examines the features, benefits, and limitations of these applications, providing a comprehensive overview of the current landscape. By analysing existing applications, the chapter aims to identify gaps and opportunities that the project seeks to address.

Chapter 3: System Methodology and Approach

Chapter 3 delves into the methodology and approach used in developing the system. It describes the overall methodology adopted for the project and outlines the general workflow procedures. This chapter also includes a system design diagram, which visually represents the architecture and design of the system, providing a clear understanding of its structure and processes.

Chapter 4: System Design

In Chapter 4, the focus shifts to the detailed design of the system. It includes a system flowchart that illustrates the operational flow of the system, helping to visualize how different components interact. The chapter also covers the database design, detailing the structure and organization of the system's database. Additionally, it presents a system block diagram, which depicts the components and their relationships. The

CHAPTER 1

chapter further specifies the component specifications and describes the functions and features of the system.

Chapter 5: System Implementation

Chapter 5 documents the implementation phase of the system. It outlines the setup requirements for both software and hardware, detailing the necessary configurations for system operation. The chapter also explains the system's operations and provides in-depth information on how each function of the system has been implemented. This section ensures that the reader understands the practical aspects of system deployment and functionality.

Chapter 6: System Evaluation and Discussion

In Chapter 6, the report evaluates the system's performance and effectiveness. This evaluation assesses how well the system meets its objectives and user needs. The discussion section analyses the results, exploring the implications of the findings and considering potential improvements. This chapter provides a critical examination of the system, highlighting its strengths and areas for enhancement.

Chapter 7: Conclusion and Recommendations

Chapter 7 concludes the report by summarizing the key findings and outcomes of the project. It reflects on the overall success of the project and its contributions. Additionally, the chapter offers recommendations for future improvements and further research, suggesting ways to build on the project's results and address any identified gaps.

CHAPTER 2 LITERATURE REVIEW

2.1 System review on existing diet and nutrition applications

i. MyFitnessPal

MyFitnessPal is a prominent diet and nutrition app known for its robust features that facilitate effective health management. With its extensive food database, users can accurately track their nutritional intake, ensuring meticulous calorie and nutrient monitoring. The integration of barcode scanning functionality further enhances the app's utility by allowing users to effortlessly input packaged foods. This streamlined approach simplifies data entry, contributing to users' adherence to their dietary goals.

One of the app's strengths lies in its goal-setting mechanisms, which enable users to set personalized targets for calorie consumption, physical activity, and weight management. The comprehensive reporting and progress tracking features empower users to visualize their journey and make informed adjustments as needed. Moreover, the app fosters a sense of community by offering a social platform for users to connect, share insights, and provide mutual support, promoting accountability and motivation.

A noteworthy feature is the innovative meal scan function that has been integrated into the app. This feature enables users to analyze their meals visually, leveraging image recognition technology. By capturing an image of their meal, users can receive real-time insights into nutritional composition, helping them make informed choices even when faced with visually complex dishes.

However, MyFitnessPal is not without limitations. While the app's extensive database is a strength, there may be instances of inaccuracies in nutritional information for user-contributed entries. Additionally, some users may find the interface overwhelming due to the multitude of features, potentially affecting the user experience, especially for newcomers. Nevertheless, MyFitnessPal remains a powerful tool for individuals seeking a comprehensive solution to managing their diet and nutrition effectively.

ii. MyNetDiary

MyNetDiary is a diet and nutrition app known for its user-friendly interface and comprehensive features that aid individuals in managing their dietary habits and promoting overall well-being. The app's strength lies in its simplicity and accessibility, making it an attractive choice for users of various backgrounds and levels of nutrition knowledge.

One of MyNetDiary's notable functions is its robust food database, which provides an extensive range of foods for users to log and track. This feature facilitates accurate calorie and nutrient tracking, ensuring that users can maintain an accurate record of their dietary intake. Moreover, the app employs barcode scanning technology to streamline data entry, making it convenient for users to log their meals.

Another strength of MyNetDiary is its integration with wearable fitness devices, allowing users to sync their physical activity data seamlessly. This function enables a holistic approach to health management, as users can track both their dietary and exercise habits within a unified platform.

Additionally, MyNetDiary offers personalized recommendations and meal planning options, providing users with guidance on achieving their health goals. This function helps users make more informed choices and tailor their dietary habits to their specific objectives, whether it's weight loss, maintenance, or muscle gain.

However, MyNetDiary is not without limitations. While the app provides a solid foundation for dietary tracking and basic guidance, some users may find the advanced features and analysis capabilities to be limited compared to other apps. Additionally, the app's focus on simplicity might not cater to individuals seeking in-depth nutritional insights or specialized dietary tracking.

In conclusion, MyNetDiary's user-friendly interface, comprehensive food database, and integration with wearable fitness devices contribute to its role as an accessible tool for individuals looking to manage their dietary habits. Its strength lies

in its simplicity and convenience, making it an ideal choice for those seeking basic nutritional tracking and guidance.

iii. Lifesum

Lifesum is a prominent diet and nutrition app that stands out for its unique approach to health management through personalized guidance and motivational features. The app's strength lies in its emphasis on creating a tailored experience for users, helping them make sustainable lifestyle changes.

A standout feature of Lifesum is its user-friendly interface that facilitates easy tracking of food intake and exercise. The app offers a barcode scanner to streamline the process of entering food items, allowing users to log meals accurately and efficiently. This functionality simplifies the user experience and encourages consistent tracking.

Lifesum's strength is evident in its focus on personalized meal plans and recommendations. The app considers users' goals, dietary preferences, and health conditions to generate customized meal plans. This feature empowers users to make informed choices that align with their objectives, whether it's weight management, muscle gain, or improved overall health.

One of Lifesum's distinctive functions is its integration with health apps and wearables, enabling users to sync data from various sources for a comprehensive overview of their well-being. This integration promotes a holistic approach to health management, giving users a complete picture of their physical activity, dietary habits, and progress.

However, Lifesum also has limitations. The app's extensive focus on personalized features may result in a steeper learning curve for some users, particularly those who prefer a more straightforward approach. Additionally, the extensive customization might lead to information overload for individuals seeking a simpler dietary tracking experience.

In conclusion, Lifesum's strength lies in its emphasis on personalized meal planning, recommendations, and integration with health data, contributing to a well-

rounded health management experience. The app's unique approach makes it appealing to users seeking a tailored approach to achieving their health and fitness goals.

iv. Cronometer

Cronometer is a widely recognized diet and nutrition app that offers a comprehensive platform for users to monitor their health and dietary habits. The app's strength lies in its meticulous tracking capabilities, allowing users to meticulously record their calorie intake, macronutrient distribution, and micronutrient consumption. By offering a detailed breakdown of nutrient values, including vitamins and minerals, Cronometer empowers users to make informed decisions about their dietary choices, ensuring optimal nutrition.

One of the notable functions of Cronometer is its emphasis on micronutrient tracking. Users can delve into their daily intake of vitamins, minerals, and other essential nutrients, providing insights into potential deficiencies or imbalances. This focus on micronutrients sets Cronometer apart as a tool that caters to individuals seeking a holistic view of their dietary health beyond just calorie counting.

The app further excels in its food recognition capabilities. Users can leverage barcode scanning and image recognition to easily log their meals, expediting the data entry process. Additionally, Cronometer provides users with the ability to set specific nutrient targets, enabling personalized goal setting tailored to individual health objectives, whether it's weight loss, muscle gain, or overall well-being.

While Cronometer offers a rich array of features, there are limitations to consider. The app's extensive nutrient tracking may be overwhelming for some users, and its focus on in-depth nutritional details might be more suitable for individuals with specific dietary requirements or health conditions. Additionally, the app's user interface may require a learning curve for new users, potentially affecting initial usability.

In summary, Cronometer serves as a valuable tool for individuals seeking precise nutrient tracking and a comprehensive understanding of their dietary intake. Its emphasis on micronutrients and food recognition functions contribute to its role in facilitating informed dietary decisions and promoting holistic health management.

2.2 Limitations of the previous studies

i. MyFitnessPal

MyFitnessPal stands as a popular dietary tracking application, equipped with a plethora of features aimed at assisting users in monitoring their nutritional intake and achieving health objectives. However, it becomes apparent that despite its comprehensive offerings, there are evident deficiencies in its coverage, notably concerning the inclusion of local Malaysian cuisine within its databases. Despite incorporating a food scanning function, these applications primarily showcase Western foods, resulting in a significant void in representing Malaysian culinary offerings. This limitation presents challenges for users endeavoring to precisely monitor their dietary intake, particularly those whose meals predominantly comprise local dishes.

While MyFitnessPal boasts a vast food database, its reliance on user-generated content may result in inaccuracies, particularly for less common or homemade foods. This lack of representation for local Malaysian dishes further exacerbates the issue, as users may struggle to find accurate nutritional information for their meals. Consequently, the effectiveness of these applications in supporting users' health goals may be compromised, particularly for those with dietary preferences or restrictions specific to Malaysian cuisine.

Moreover, while MyFitnessPal offers goal-setting functionalities, users may find the recommendations and goals overly generalized, lacking the personalization necessary to effectively meet their individual needs. This lack of customization can diminish the user experience and hinder users' ability to achieve their health objectives. Additionally, advanced features such as customized macronutrient goals and meal planning are often restricted to premium subscriptions, potentially excluding users who rely on the free version of the app.

In light of these limitations, there is a clear opportunity to develop a more tailored dietary tracking application that addresses the specific needs of Malaysian users. By prioritizing the inclusion of local Malaysian foods in its database and offering personalized goal-setting functionalities, such an application could better support users in monitoring their nutritional intake and achieving their health goals. Additionally,

ensuring accessibility to advanced features without requiring a premium subscription would enhance the inclusivity and effectiveness of the app for all users.

ii. MyNetDiary

MyNetDiary stands out as a comprehensive dietary tracking app that provides an intuitive interface and an array of features for individuals striving to manage their nutrition and fitness. Its user-friendly design is a strength; however, some users have raised concerns about difficulties encountered when manually entering custom foods or recipes. This might impact user engagement and discourage those seeking a more streamlined input process. Similar to other applications, MyNetDiary offers biometric tracking capabilities, though it may not encompass all metrics or match the accuracy of dedicated fitness wearables. Furthermore, the app's community features are comparatively limited, potentially affecting the extent of social support and interaction accessible to users.

iii. Lifesum

Lifesum, with its emphasis on health and wellness, provides a diverse range of features to support users in their dietary and lifestyle pursuits. The app's focus on mindfulness and curated recipes introduces a unique aspect to its offerings. However, it's crucial to acknowledge that Lifesum's nutritional analysis may not be as exhaustive as that of its counterparts, leaving users with a less comprehensive understanding of their nutrient intake. Additionally, the app's barcode scanning feature, though valuable, has been criticized for its accuracy, impacting the overall reliability of nutrient tracking. It's worth noting that the app's community size might be smaller compared to other platforms, potentially influencing the degree of peer support and engagement accessible to users. It's also important to mention that some of Lifesum's premium features, like mindfulness exercises and recipes, require a subscription, potentially limiting access for non-subscribing users.

iv. Cronometer

Cronometer sets itself apart with its intricate nutrient tracking capabilities, furnishing users with an exhaustive analysis of their dietary consumption. However, this level of detail might prove overwhelming for individuals seeking a more straightforward approach to nutrition tracking. The complexity of the app could potentially lead to a

learning curve for newcomers, potentially impacting engagement during the initial stages. Unlike some competing apps, Cronometer lacks social community features, which might affect users seeking support and motivation from their peers. Notably, the app's integration of expert consultations is commendable; nevertheless, the availability of such consultations could be limited or come with additional costs, rendering them inaccessible to certain users.

2.3 Compare and contrast of the previous study

The table below shows the comparison between MyFitnessPal, MyNetDiary, Lifesum, and Cronometer.

Table 2.3.1: Table of comparison between MyFitnessPal, MyNetDiary, Lifesum, and Cronometer

Feature	MyFitnessPal	MyNetDiary	Lifesum	Cronometer	NutriLife
Food Scanning	Yes	No	No	No	Yes
Personalized meal recommendation	No	No	No	No	Yes
Streak counter	No	No	No	No	Yes
Simplified interface	No	No	No	No	Yes
Remaining Days for goals	No	No	Yes	No	Yes
Food insights	Yes	Yes	Yes	No	Yes
Meal planning	Yes	Yes	No	No	Yes
AI Fitness tracking	No	No	No	No	Yes
Malaysian local food database	No	No	No	No	Yes

CHAPTER 3 SYSTEM METHODOLOGY/APPROACH

3.1 Methodology and General Workflow Procedures

After a comprehensive analysis of software development methodologies, this proposed project is chosen to implement Personal Extreme Programming (PXP) in this project. PXP shares principles and practices with the well-known Agile methodology Extreme Programming (XP). XP is known for its collaborative approach where developers work in pairs, enabling continuous code review, knowledge sharing, and mutual support for top-notch results. However, traditional XP practices are tailored for teams with multiple members.

In our project, the key challenge lies in the fact that it consists of a single developer. Traditional XP, with its emphasis on pair programming, is not directly applicable. Therefore, we're opting for PXP, which is rooted in XP principles and crafted to meet our specific needs.

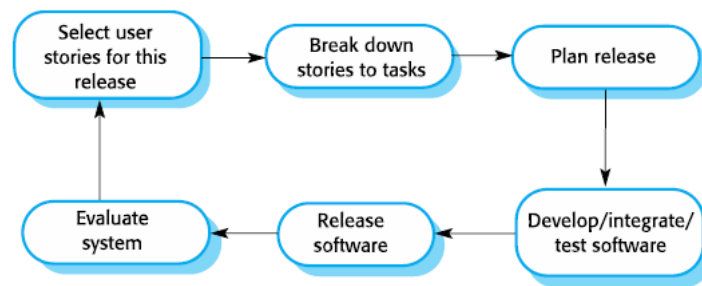


Figure 3.1.1: The extreme programming release cycle

Extreme Programming (XP) operates by expressing requirements as scenarios, referred to as user stories, which are then translated into a series of tasks. Programmers collaborate in pairs, creating tests for each task before proceeding with the actual coding. To integrate new code into the system, all tests must pass successfully. XP follows a frequent release cycle with short intervals between system releases. According to the figure above, it provides a visual representation of the XP process for generating increments of the ongoing system development.

3.1.1 User stories

In the proposed project, there will be only one user to be concentrated which is the user who will use NutriLife.

As a user, I can:

1. **Create a New Account:** Easily sign up for a new account using a user-friendly registration process that requires a unique username, secure password, and valid email address.
2. **Personalize My Account:** Customize my account with personal information, dietary preferences, and fitness goals to receive tailored recommendations and insights.
3. **Log In Securely:** Access my account through a secure login process that verifies my credentials and ensures my personal data is protected.
4. **Monitor My Posture:** Utilize the AI Fitness Tracking Component to continuously track and analyse my posture during workouts, ensuring that I maintain proper form and alignment.
5. **Receive Real-Time Feedback:** Get instant feedback on my exercise technique, helping me correct any deviations and improve my performance effectively.
6. **Enhance Workout Efficiency:** Benefit from personalized recommendations based on my posture analysis, optimizing my workouts for better results and reducing the risk of injury.
7. **Reduce Injury Risk:** Utilize the component's posture correction features to minimize the chances of workout-related injuries by ensuring that I perform exercises correctly.
8. **Track Progress Over Time:** View historical data on my posture and exercise performance, allowing me to track improvements and adjust my workout routines as needed.
9. **Track My Progress with a Streak Counter:** Monitor my activity and achievements with a streak counter that helps keep me motivated by showing how consistently I'm meeting my fitness and dietary goals.
10. **Scan and Identify Food:** Capture images of my meals using the Food Scanning Component or upload photos from my gallery to get accurate identification and nutritional information for each food item.

11. **Record and Track Food Intake:** Record the foods I snap or upload, integrating this data into my daily food intake tracking to manage my nutrition effectively.
12. **Receive Nutritional Insights:** Access detailed nutritional information, including calorie counts and macronutrient breakdowns, to make informed dietary choices.
13. **Get Recipe Recommendations:** Explore and receive suggestions for simple and easy-to-cook dishes based on the food database, helping me plan my meals effectively.
14. **Maintain a Balanced Diet:** Utilize the food and fitness data to ensure that my diet complements my exercise routine, supporting my health and fitness objectives.

3.1.2 Tasks

The user stories have been proposed in section 4.1.1 (User stories). Hence, in this section, the user stories will be broken down into a few sprints.

Sprint goal 1

To Implement the user account management features, including creating new accounts, personalized account settings, and secure login functionality.

Sprint backlog 1

- As a NutriLife's user, I want to experience a responsive and visually appealing user interface during the registration process, so that I can enjoy a consistent and delightful experience across various devices. (2)
- As a NutriLife's user, I want to be guided through a seamless and understandable onboarding process, so that I can effortlessly provide my basic information and preferences when I launch the app for the first time. (3)
- As a NutriLife's user, I want to create a unique username during registration, so that I can establish a personalized account that reflects my individuality. (1)
- As a NutriLife's user, I want to set up a distinct password during registration, so that I can ensure the security of my account and gain access to it as needed. (1)
- As a NutriLife's user, I want to provide my email address for account verification, so that I can regain access to my accounts in case I forget my password, enhancing account recovery. (2)
- As a NutriLife's user, I want to receive the email verification link upon registration, so that I can successfully verify my account and complete the registration process. (1)
- As a NutriLife's user, I want to receive a confirmation message after registration, so that I can ensure that my account activation is confirmed and finalized. (1)
- As a NutriLife's user, I want to log in securely with my login credentials, so that I can safeguard my account from unauthorized access and maintain the privacy of my personal information. (3)
- As a NutriLife's user, I want multiple login attempts, so that I have the flexibility to correct mistakes without being locked out of my account. (3)

CHAPTER 3

- As a NutriLife's user, I want my account to be temporarily locked after multiple failed login attempts, so that I can prevent unauthorized users from gaining access to my account through brute force attacks. (3)
- As a NutriLife's user, I want to provide my email address for the password recovery option, so that I can securely verify my identity and recover my account access. (3)
- As a NutriLife's user, I want a password recovery option available, so that I have a reliable way to regain access to my account if I ever forget my password. (3)
- As a NutriLife's user, I want a user profile customization interface, so that I can tailor my profile to reflect my preferences and personality. (3)
- As a NutriLife's user, I want to upload profile pictures, so that I can replace the default system profile image and personalize my account appearance. (2)
- As a NutriLife's user, I want a "Gender" option to select gender for my user profile, so that I can choose whether to share my gender information. (2)
- As a NutriLife's user, I want validation checks on the information I input so that I can ensure accuracy and prevent errors in my account details. (2)
- As a NutriLife's user, I want the option to "Modify" my personal information, so that I can correct any inaccuracies and keep my profile up to date. (2)
- As a NutriLife's user, I want to receive a "Modification successful" message, so that I can be assured that my user profile has been successfully updated. (1)
- As a NutriLife's user, I want the option to "Save" my updated personal information, so that I can preserve the latest changes and keep my profile current. (3)
- As a NutriLife's user, I want to receive a "Save successful" message, so that I can be informed that the changes I made to my profile were successfully preserved. (1)

Estimated velocity: 42 story points

CHAPTER 3

Sprint goal 2

To build and integrate the food scanning feature to recognize food items and provide nutritional insights.

Sprint backlog 2

- As a NutriLife's user, I want to scan images of my food or upload photos from my gallery, so that I can get accurate identification and detailed nutritional information about what I am eating. (1)
- As a NutriLife's user, I want to track the nutritional content of my meals, so that I can make informed dietary choices and manage my nutrition effectively. (2)
- As a NutriLife's user, I want to view detailed nutritional insights based on the food I scan, so that I can understand the calorie counts and macronutrient breakdowns of my meals. (1)
- As a NutriLife's user, I want to save scanned food items to a personal food log, so that I can easily review my food history and track my intake over time. (2)
- As a NutriLife's user, I want to receive suggestions for healthier food options based on my scanned items, so that I can make better dietary choices. (3)
- As a NutriLife's user, I want to track my fitness and dietary progress with a streak counter, so that I can stay motivated and monitor my consistency in meeting my health goals. (2)
- As a NutriLife's user, I want to receive personalized recipe recommendations based on my food intake and preferences, so that I can easily plan and prepare meals that support my dietary goals. (2)
- As a NutriLife's user, I want to view suggested recipes in a user-friendly format, so that I can quickly find and prepare meals that fit my nutritional needs. (3)
- As a NutriLife's user, I want to track my daily streaks for both fitness and nutrition, so that I can visualize my progress and stay motivated to achieve my goals. (2)
- As a NutriLife's user, I want to set personal goals and track my progress towards them, so that I can stay focused and measure my success over time. (3)

Estimated velocity: 21 story points

CHAPTER 3

Sprint goal 3

To develop an AI-driven posture tracking system using MediaPipe and computer vision, so that NutriLife users can receive real-time feedback on their posture during workouts, ensuring proper form and reducing the risk of injury.

Sprint backlog 3

- As a NutriLife's user, I want to monitor my posture during workouts using the AI fitness tracking component, so that I can receive real-time feedback to ensure I am exercising with proper form and reduce the risk of injury. (1)
- As a NutriLife's user, I want to receive alerts for any detected posture issues during my workouts, so that I can make immediate corrections and avoid injuries. (2)
- As a NutriLife's user, I want to get personalized workout recommendations based on my posture analysis, so that I can enhance my workout efficiency and effectiveness. (2)
- As a NutriLife's user, I want to view historical data on my posture and exercise performance, so that I can track improvements and adjust my workout routines as needed. (3)
- As a NutriLife's user, I want to have an intuitive interface for accessing and reviewing posture feedback and workout recommendations, so that I can easily understand and act on the guidance provided. (3)
- As a NutriLife's user, I want to receive a summary of my posture analysis after each workout, so that I can monitor my progress and make informed decisions about my exercise regimen. (2)

Estimated velocity: 13 story points

3.1.2 Model Training

3.1.2.1 Food recognition model training

After planning and breaking down the user stories, the next crucial step in developing the food recognition system is model training. This phase ensures that the AI component accurately identifies and classifies various types of Malaysian food based on the collected dataset.

The process begins with data collection, where a comprehensive dataset of 20,000 images is gathered, representing eight distinct classes of Malaysian food. Each class is supported by 2,500 images to ensure a balanced and robust dataset. The collected images are then preprocessed to enhance their quality. This involves removing low-quality or irrelevant images, resizing, and normalizing them to maintain consistency across the dataset. Effective preprocessing is crucial for improving the training efficiency and performance of the model.

Name	Date modified	Type	Size
half_boiled_egg	09/04/24 1:21 AM	File folder	
kaya_toast	09/04/24 1:19 AM	File folder	
kuey_teow	09/04/24 1:21 AM	File folder	
laksa	09/04/24 1:21 AM	File folder	
nasi_lemak	09/04/24 1:21 AM	File folder	
popiah	09/04/24 1:22 AM	File folder	
roti_canai	09/04/24 1:22 AM	File folder	
satay	09/04/24 1:21 AM	File folder	

Figure 3.1.2.1 Eight distinct classes of Malaysian Food




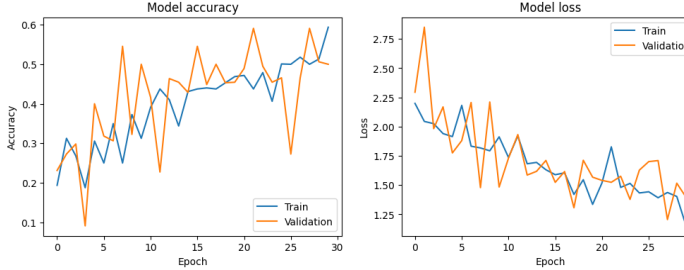
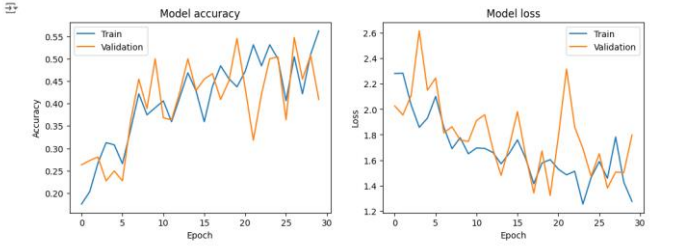
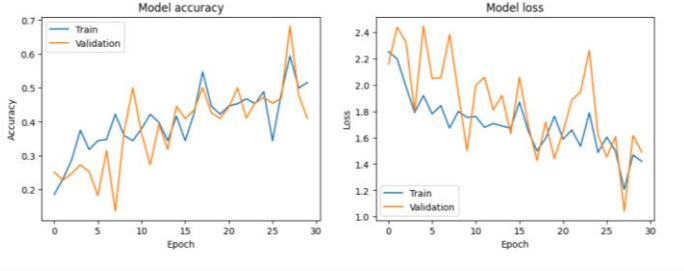
Figure 3.1.2.2: Example datasets collected for Nasi Lemak class

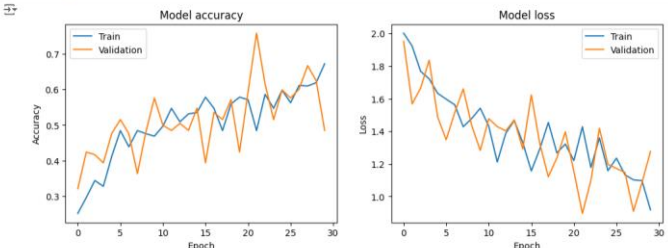
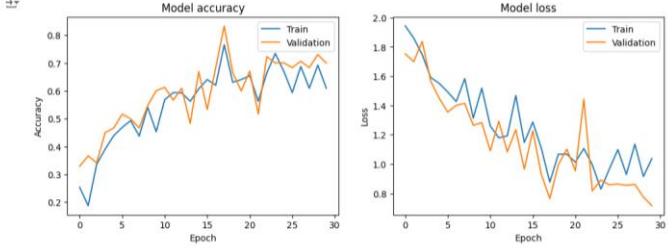
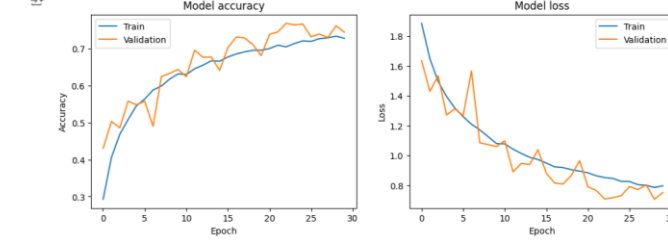
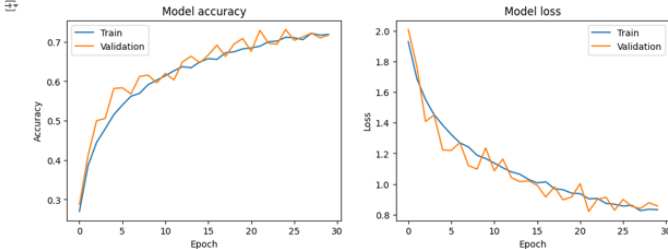
For this project, a Convolutional Neural Network (CNN) was chosen for its demonstrated effectiveness in image classification tasks. The CNN is designed to process the dataset and accurately classify food images. During the training phase, key parameters, including batch size, are adjusted to optimize the model's performance, as detailed in Table 3.1.2. The model undergoes multiple epochs to learn from the data, with hyperparameters such as learning rate and optimization algorithms being fine-tuned to improve accuracy.

Table 3.1.2.3 illustrates that factors like the number of datasets, dense layers, and batch size significantly impact the model's performance, particularly its accuracy. Each of these factors is tested at least 10 times to ensure robust results and to prevent issues like overfitting or underfitting. In the graphical representation, the blue line represents training datasets, while the orange line represents validation datasets. A smaller gap between the validation dataset (orange line) and the training dataset indicates better generalization. This suggests that the model is capable of performing well on new, unseen data, rather than just on the data it was trained with.

Next, the blue lines indicated the performance metrics of the training dataset throughout the model's training process. These lines illustrate how the model's accuracy evolves as it learns from the training data over multiple epochs. The blue lines help visualize the model's progress and effectiveness in learning from the training data, showing improvements or fluctuations in accuracy at each stage of training. A steady and upward slope in the blue lines indicates successful learning and increasing accuracy on the training set as shown in the last two columns of the table 3.1.2.3.

Table 3.1.2.3: Different batch size and number of datasets affecting the model's performance

Batch size	Number of datasets	Model's performance
20	10,000	<pre data-bbox="635 387 1321 454"> plt.xlabel('Epoch') plt.ylabel('Loss') plt.legend(['Train', 'Validation']) plt.show() </pre>  <p data-bbox="635 734 742 757">save model</p> <pre data-bbox="635 779 1321 824"> [56] model.save('food_classifier_model.h5') </pre> <p data-bbox="635 806 1321 828">WARNING:absl:You are saving your model as an HDF5 file via 'model.save()' or 'keras.saving.save_model()'. This file format is co</p>
32	10,000	
32	11,000	<pre data-bbox="635 1149 1321 1238"> [13] plt.plot(history.history['val_loss']) plt.title('Model loss') plt.xlabel('Epoch') plt.ylabel('Loss') plt.legend(['Train', 'Validation']) plt.show() </pre> 
64	11,000	

<p>64</p>	<p>15,000</p>	<pre>plt.legend(['Train', 'Validation']) plt.show()</pre> 
<p>64</p>	<p>15,000 (Add 2 more dense layer)</p>	<pre>plt.legend(['Train', 'Validation']) plt.show()</pre> 
<p>64</p>	<p>20,000</p>	<pre>plt.show()</pre>  <p>save model</p> <pre>[39] model.save('nutr111fe2.h5')</pre> <p>WARNING:absl:You are saving your model as an HDF5 file via 'model.save()' or 'keras.saving.save_model(model)'. This file format is con</p> <p>0s completed at 5:40AM</p>
<p>64</p>	<p>20,000 (Add 2 more dense layer)</p>	<pre>plt.show()</pre> 

Evaluation of the model’s performance involves generating accuracy metrics and confusion matrices. Accuracy provides an overall measure of how well the model classifies images into the correct food categories. The confusion matrix offers a more detailed view, showing true positives, false positives, true negatives, and false negatives for each class. This matrix helps identify specific areas where the model may be underperforming and provides insights for improvement as shown in figure 3.1.2.4.

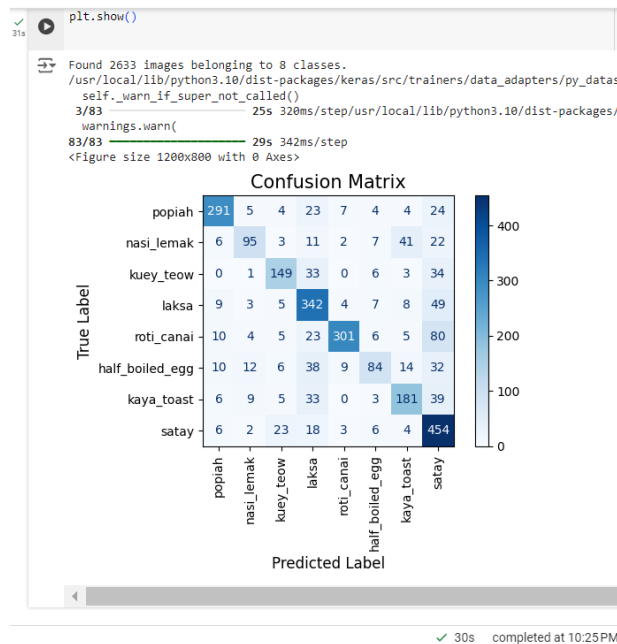


Figure 3.1.2.4 Confusion matrix for food recognition model of NutriLife

Graphs depicting model accuracy and validation metrics are used to visualize the model's performance throughout the training process. The accuracy graph illustrates how the model's classification performance evolves over time, with training accuracy typically increasing as the model learns. The validation accuracy graph helps assess how well the model generalizes to new, unseen data, highlighting any discrepancies between training and validation performance that could indicate overfitting or underfitting as shown in table 3.1.2.3.

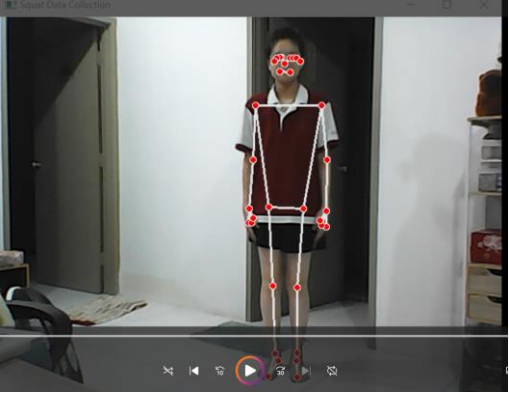

Once the model reaches satisfactory performance levels, indicated by high accuracy and a favorable confusion matrix, it is integrated into the NutriLife application. Post-deployment, the model is continuously monitored and evaluated based on user feedback and performance data to ensure its ongoing accuracy and effectiveness in providing reliable food identification and nutritional insights.

3.1.2.2 AI Fitness model training

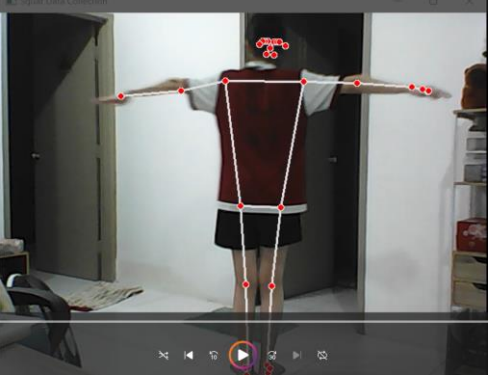
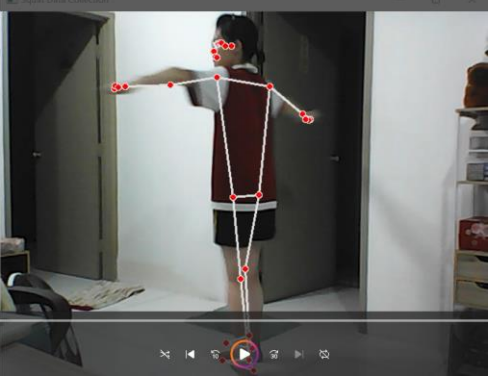

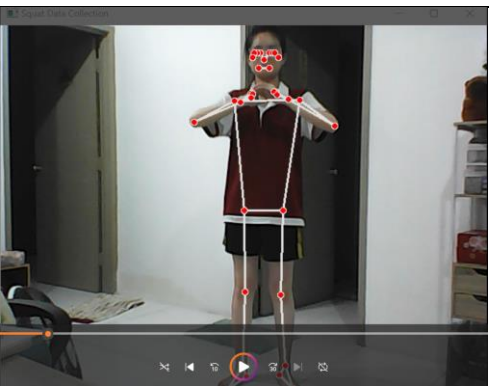
After planning and breaking down the user stories, the next critical phase is model training for the AI-driven posture tracking system. This step is essential for ensuring the AI component accurately analysis and provides feedback on users' posture during workouts.

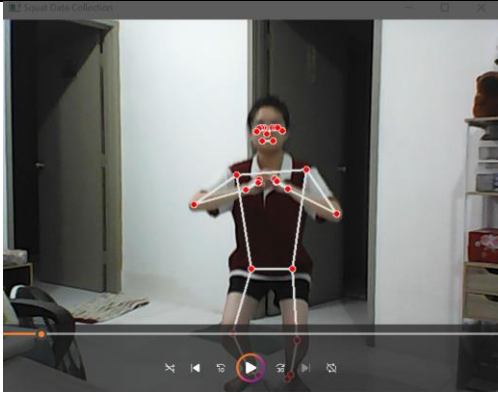

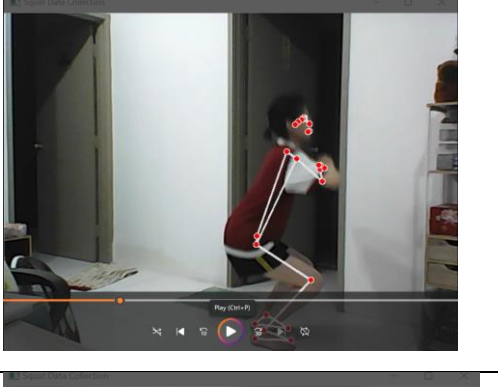

The process begins with data collection, where a diverse dataset of workout videos and images is gathered as shown in table 3.1.2.2.0. This dataset should represent various postures, exercises, body types, and movement ranges to ensure comprehensive model training. Once collected, the data is meticulously annotated with key points related to posture, such as joint positions and body alignment, to facilitate accurate analysis.

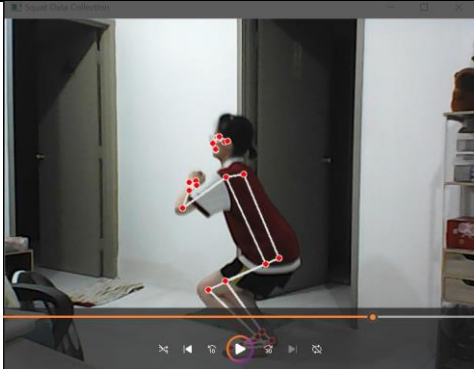
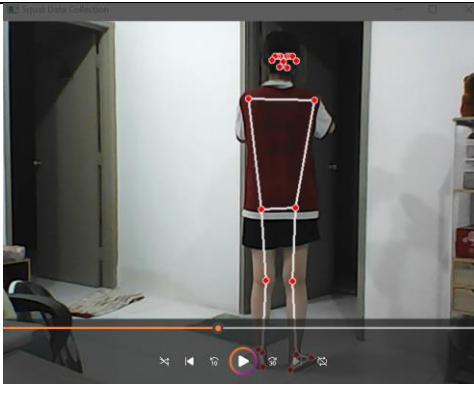
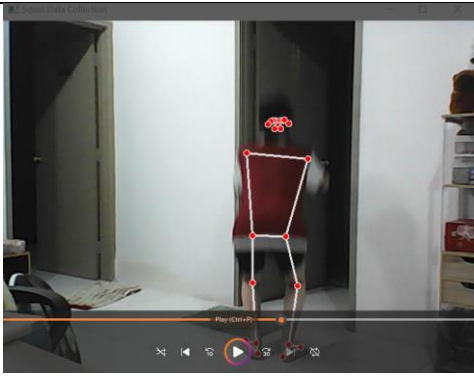
Table 3.1.2.2.0: Squat data collection

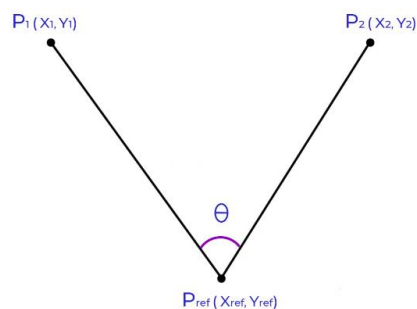
Squat data collection description	Screenshot of squat data collection video
Extract coordinates key landmarks: hip, knee, ankle (front view)	
Extract coordinates key landmarks: hip, knee, ankle (front view)	

CHAPTER 3

<p>Extract coordinates key landmarks: hip, knee, ankle</p> <p>(back view)</p>		
<p>Extract coordinates key landmarks: hip, knee, ankle</p> <p>(side view)</p>		
<p>Extract coordinates key landmarks: hip, knee, ankle</p> <p>(side view)</p>		
<p>Compute the angle at knee using the calculation_angle() as shown in figure 3.1.2.2.1, which calculates the angle between the hip, knee, and ankle points</p> <p>(front view - up)</p>		

<p>Compute the angle at knee using the <code>calculation_angle()</code> as shown in figure 3.1.2.2.1, which calculates the angle between the hip, knee, and ankle points</p> <p>(front view - down)</p>	
<p>Compute the angle at knee using the <code>calculation_angle()</code> as shown in figure 3.1.2.2.1, which calculates the angle between the hip, knee, and ankle points</p> <p>(side view - up)</p>	
<p>Compute the angle at knee using the <code>calculation_angle()</code> as shown in figure 3.1.2.2.1, which calculates the angle between the hip, knee, and ankle points</p> <p>(side view - down)</p>	
<p>Compute the angle at knee using the <code>calculation_angle()</code> as shown in figure 3.1.2.2.1, which calculates the angle between the hip, knee, and ankle points</p> <p>(side view - up)</p>	

<p>Compute the angle at knee using the <code>calculation_angle()</code> as shown in figure 3.1.2.2.1, which calculates the angle between the hip, knee, and ankle points</p> <p>(side view - down)</p>	
<p>Compute the angle at knee using the <code>calculation_angle()</code> as shown in figure 3.1.2.2.1, which calculates the angle between the hip, knee, and ankle points</p> <p>(back view - up)</p>	
<p>Compute the angle at knee using the <code>calculation_angle()</code> as shown in figure 3.1.2.2.1, which calculates the angle between the hip, knee, and ankle points</p> <p>(back view - down)</p>	

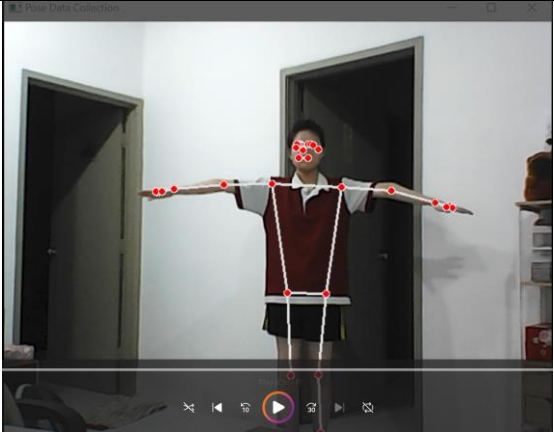
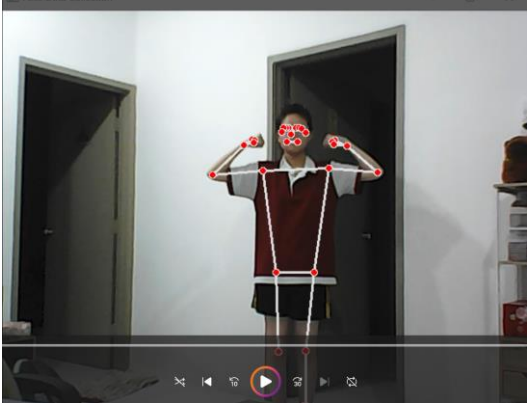
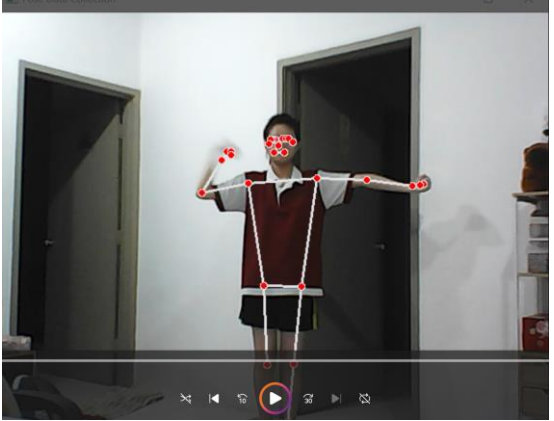


The equation is given by:

$$\theta = \arccos \frac{\vec{P_{1ref}} \cdot \vec{P_{2ref}}}{|\vec{P_{1ref}}| \cdot |\vec{P_{2ref}}|}$$

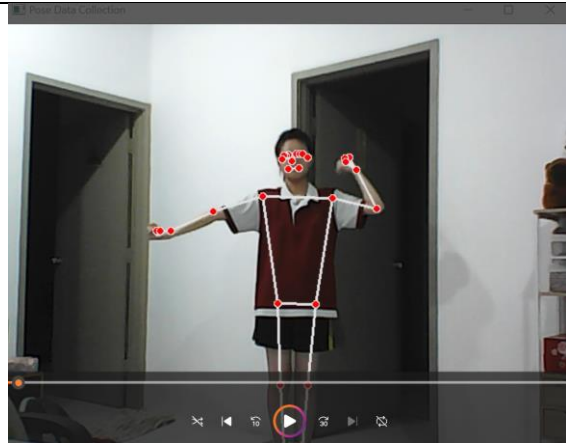
Figure 3.1.2.2.1: Calculation_angle function's equation

Table 3.1.2.2.1: Bicep curl data collection

Bicep curl data collection description	Screenshot of bicep curl data collection video
Extract coordinates key landmarks: shoulder, elbow, wrist (front-view)	 A screenshot from a video titled "Pose Data Collection" showing a person in a red vest and black shorts standing in a doorway. The person's arms are extended horizontally to the sides. Red circular markers are placed on the head, shoulders, elbows, and wrists. White lines connect these markers to form a skeletal model of the upper body. The video player interface is visible at the bottom.
Extract coordinates key landmarks: shoulder, elbow, wrist (front-view-both hands)	 A screenshot from a video titled "Pose Data Collection" showing the same person in the same doorway. The person's hands are raised towards their shoulders. Red circular markers are placed on the head, shoulders, elbows, and wrists. White lines connect these markers to form a skeletal model. The video player interface is visible at the bottom.
Extract coordinates key landmarks: shoulder, elbow, wrist (front-view – right-hand)	 A screenshot from a video titled "Pose Data Collection" showing the same person in the same doorway. The person's right hand is raised towards their shoulder, while the left arm is extended horizontally. Red circular markers are placed on the head, shoulders, elbows, and wrists. White lines connect these markers to form a skeletal model. The video player interface is visible at the bottom.

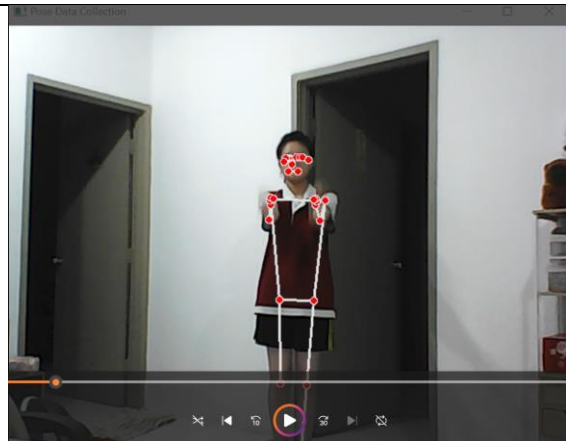
Extract coordinates key landmarks:
shoulder, elbow, wrist

(front-view-left hand)



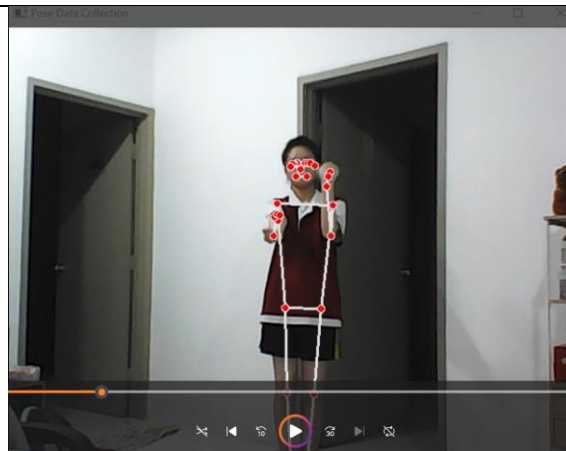
Extract coordinates key landmarks:
shoulder, elbow, wrist

(front-view-both hands put in front)



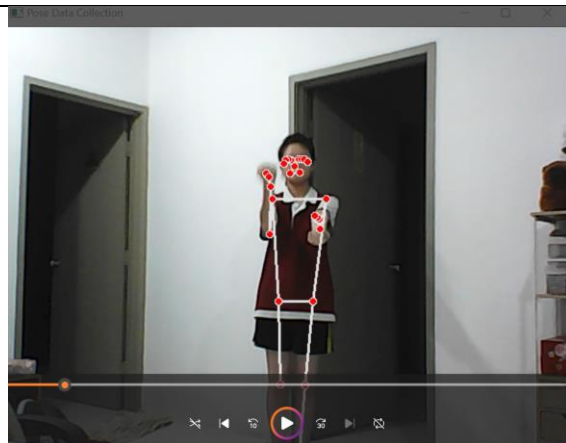
Extract coordinates key landmarks:
shoulder, elbow, wrist

(front-view- left hand lift up)



Extract coordinates key landmarks:
shoulder, elbow, wrist

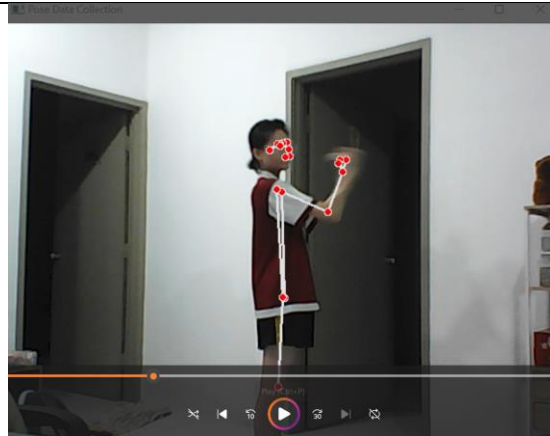
(front-view- right hand lift up)



CHAPTER 3

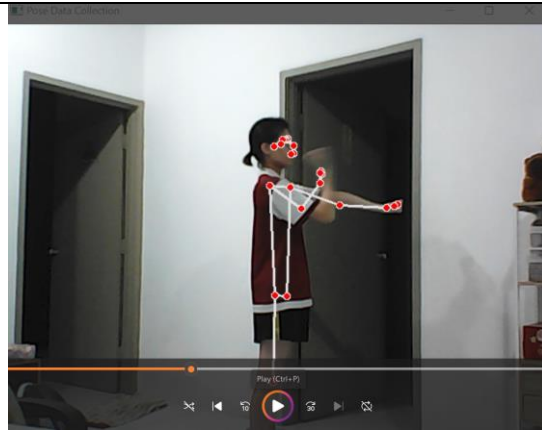
Extract coordinates key landmarks:
shoulder, elbow, wrist

(side-view- both hands lift up)



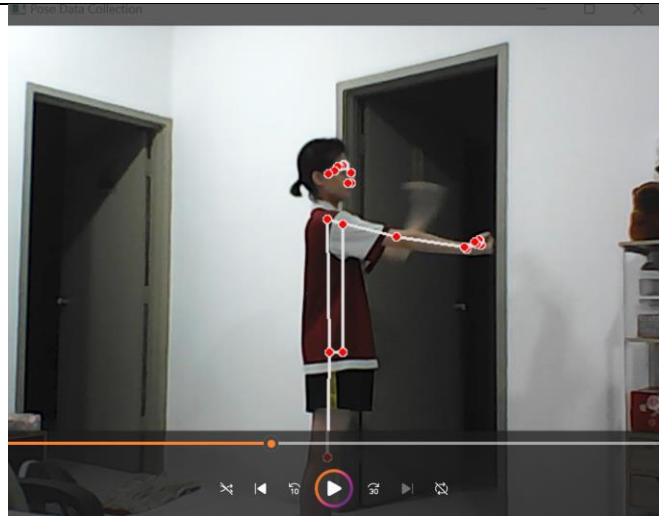
Extract coordinates key landmarks:
shoulder, elbow, wrist

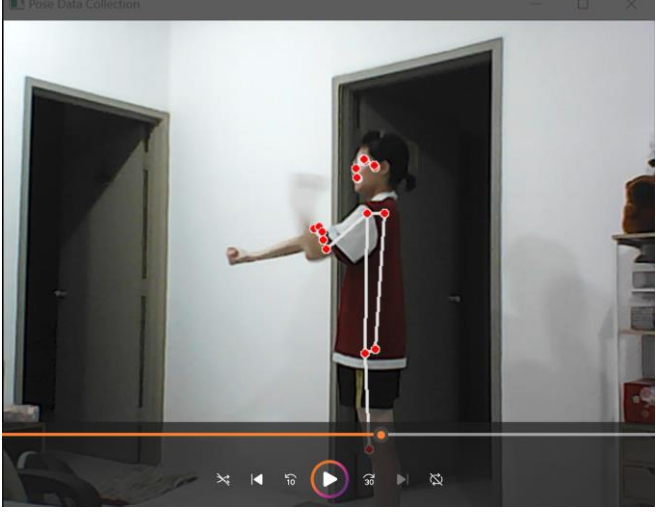
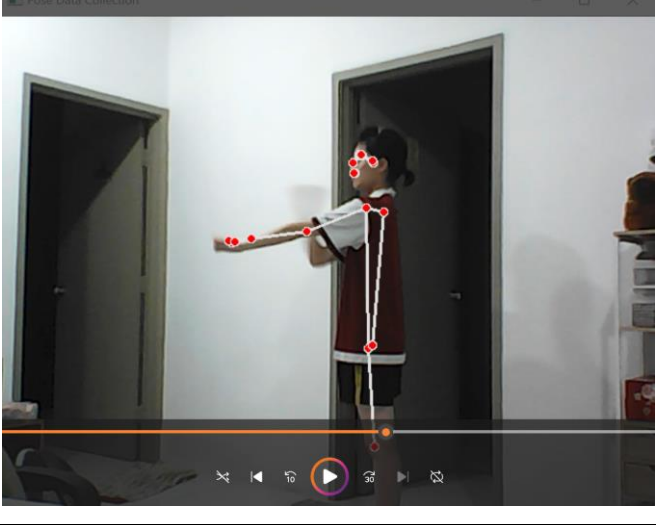
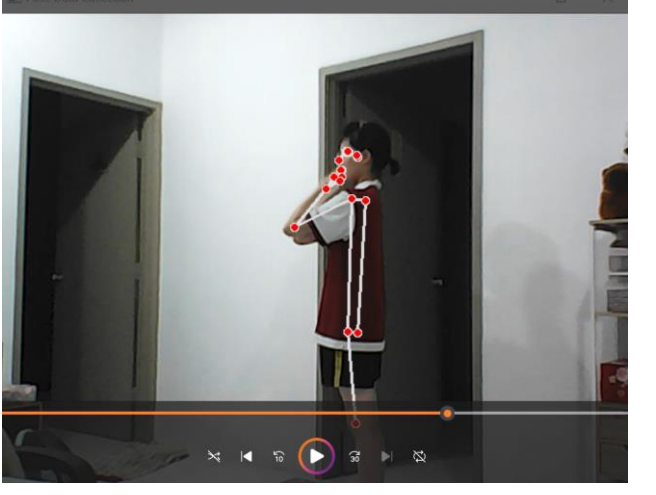
(side-view- right hand lift up)



Extract coordinates key landmarks:
shoulder, elbow, wrist

(side-view- left hand lift up)



<p>Extract coordinates key landmarks: shoulder, elbow, wrist</p> <p>(side-view- left hand lift up)</p>	
<p>Extract coordinates key landmarks: shoulder, elbow, wrist</p> <p>(side-view- right hand lift up)</p>	
<p>Extract coordinates key landmarks: shoulder, elbow, wrist</p> <p>(side-view- both hands lift up)</p>	

Next, the data undergoes pre-processing, which includes cleaning and normalizing images, resizing them for consistency, and ensuring uniform lighting conditions. This step also involves splitting the dataset into training, validation, and test sets to enable accurate evaluation of the model's performance.

Choosing the appropriate model is crucial in this phase. For posture analysis, the pipeline for MediaPipe pose has been implemented in this proposed project as shown in figure 3.1.2.2.2. With this pipeline, it will first locate the person within the frames and the tracker subsequently predicts the pose landmarks and segmentation marks within the frame.

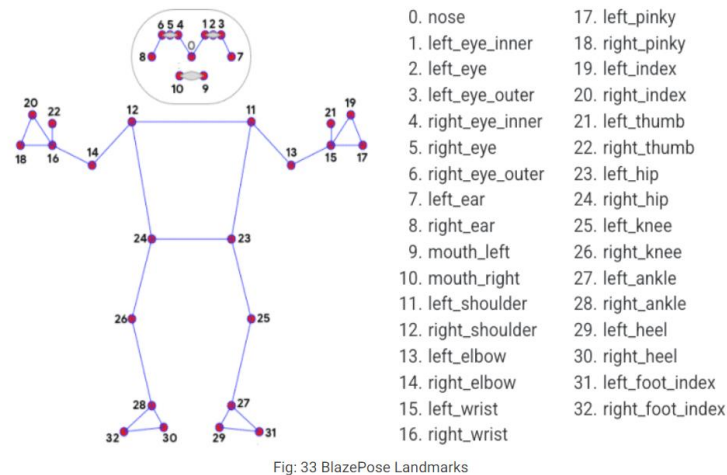


Figure 3.1.2.2.2: Pose landmarks

The training process involves feeding the pre-processed data into the model. During this phase, techniques such as transfer learning may be used to accelerate the training by leveraging pre-trained models. It is essential to monitor the training process closely, adjusting hyperparameters as needed to improve the model's accuracy and performance.

Following training, the model's effectiveness is evaluated using the validation set, assessing metrics like accuracy, precision, and recall. Testing on a separate test set ensures the model generalizes well to new data and performs reliably across different scenarios.

Refinement of the model may be necessary based on evaluation results. This could involve adjusting training strategies or incorporating additional data to address any issues identified during testing. Iterative refinement helps in enhancing the model's ability to provide real-time, accurate posture feedback.

3.2 System Design

3.2.1 System Architecture Design

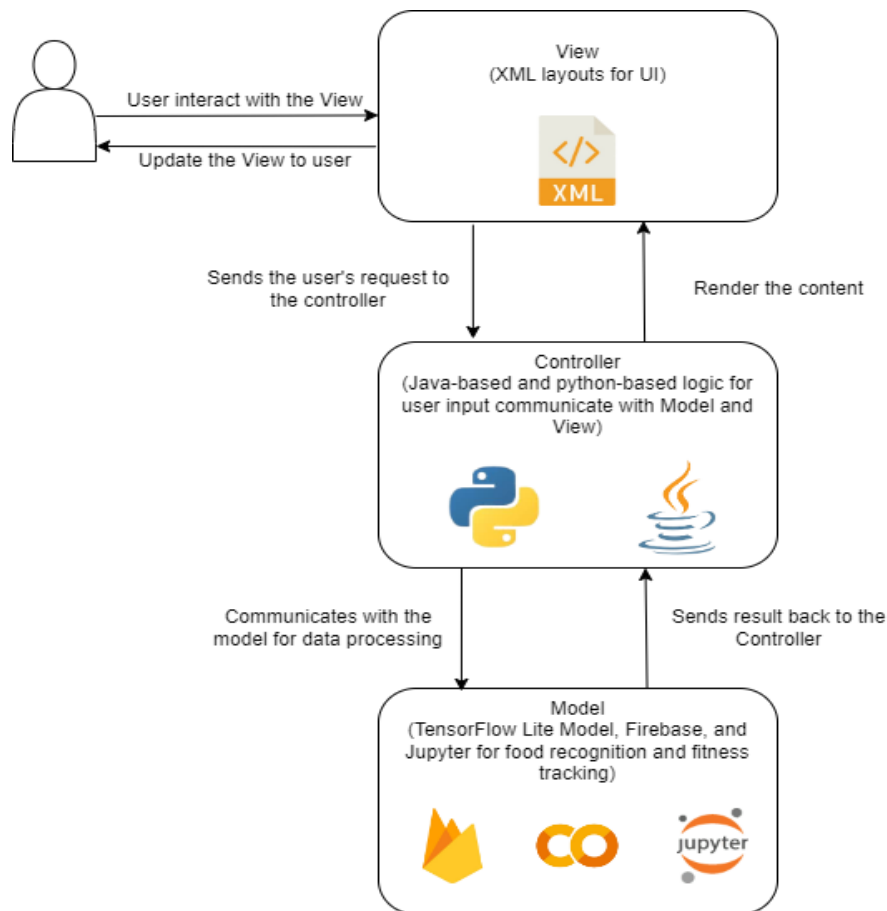


Figure 3.2.1.0: System Architecture Design

The system architecture NutriLife is implemented with Model-View-Controller (MVC) architecture, ensuring a clear division between the user interface, logic, and data processing layers. This design allows for efficient communication between the app's components, ensuring smooth functionality and user interaction.

The View serves as the user interface of the application, using XML layouts for Android development. It is responsible for displaying the graphical elements that users interact with, such as buttons, text inputs, and image viewers. The View also captures the user's inputs, such as taking photo a food item or engaging in fitness activities such as start squat counting, and forwards these requests to the Controller. After the Controller processes the request, the View updates the UI with the appropriate results, whether it is showing recognized food items, nutritional information, or posture feedback during exercise.

CHAPTER 3

The Controller acts as a mediator between the View and the Model. It handles the app's logic, receiving input from the user via the View and processing it. The Controller, implemented using Java for Android functionality and Python for machine learning integration, communicates with the Model to request data processing tasks. Once the Model completes its operations, the Controller retrieves the results and sends them back to the View for display. This component ensures that the data flow between the UI and the underlying data processing models is seamless and efficient.

The Model is responsible for the core data processing and machine learning tasks in the system. It utilizes TensorFlow Lite to run the food recognition models on the mobile device and uses Firebase for cloud-based data storage. The machine learning models, which are developed and fine-tuned using Google Colab, handle tasks like classifying images of Malaysian food and Jupyter for analyzing posture for exercises like squats. The Model processes the data received from the Controller and returns the results, whether it is a predicted food item or feedback on exercise posture. This architecture ensures that all machine learning tasks are handled effectively while maintaining smooth user interaction.

3.2.2. Use Case Diagram

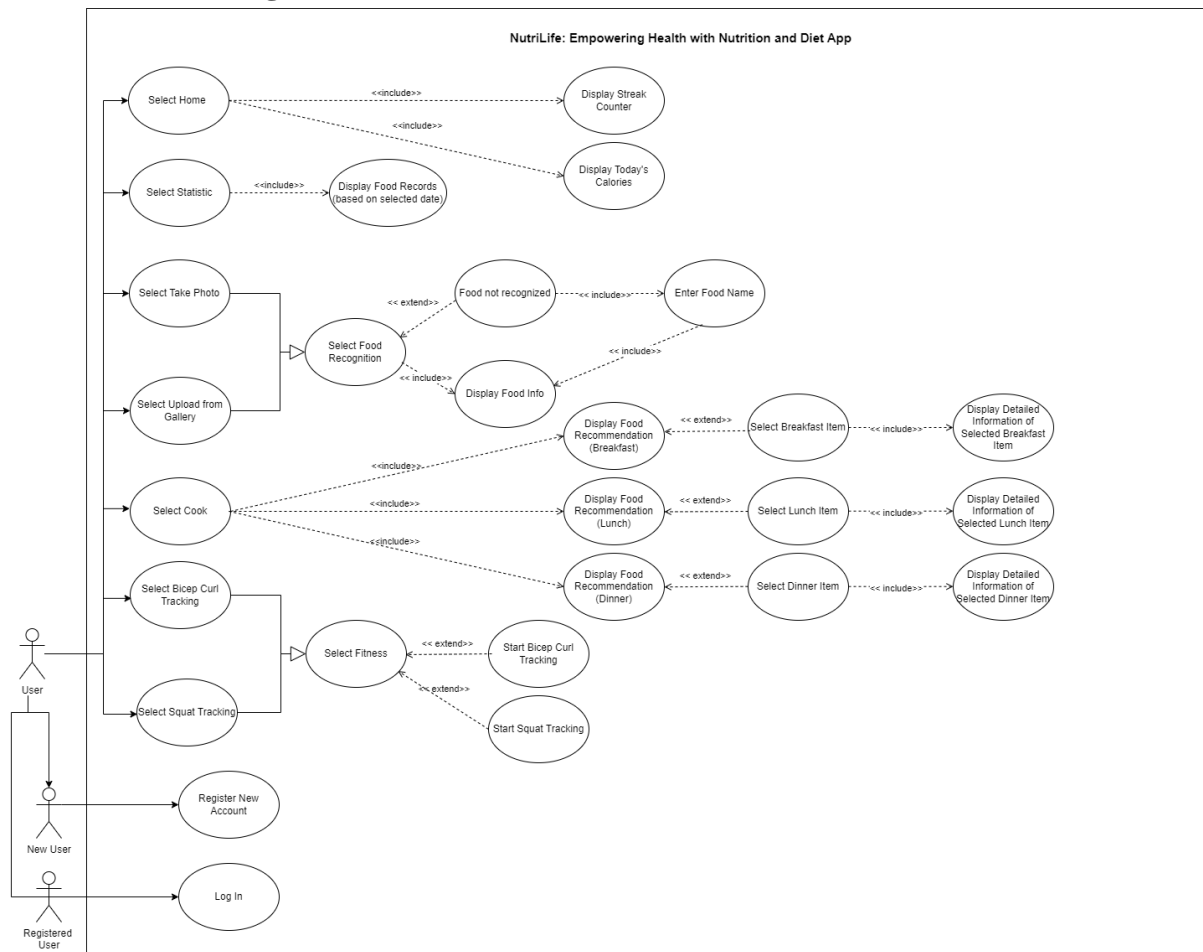


Figure 3.2.2.0: Use Case Diagram for NutriLife

User Registration and Login

For new users, NutriLife provides an option to Register a New Account, enabling them to input the necessary details and create an account. Upon successful registration, the user becomes a Registered User and can log in to access the app’s personalized features. The Log In use case supports registered users in securely accessing their accounts, allowing them to view their saved food records, fitness data, and meal recommendations.

Home Screen Interaction

According to the figure as shown above, when a registered user selects the **home** option, the app displays personalized information such as daily calorie intake and streak counters, allowing users to track their progress over time. This use case is designed to provide immediate feedback, reinforcing the user's consistency in tracking their meals and fitness activities. The user can view a streak counter that encourages regular usage and a calorie count for the day to help with diet monitoring.

Food Recognition and Meal Tracking

Another core use cases are the Take Photo feature, where users capture a photo of their meal for recognition. Upon selecting this option, the app uses an AI-powered food recognition system to identify the food item. If the food is successfully recognized, the app displays detailed nutritional information in the Food Info section, including calorie content, macronutrients, and other dietary data. If the food is not recognized, the user can manually enter the food's name, and the system will then display the associated nutritional information. Similarly, users can upload images from their gallery through the Upload from Gallery feature, enabling them to retrieve the same recognition and nutrition details.

Cook

According to the figure 3.2.2.0, when the user selects the Cook option, they are presented with food recommendations for breakfast, lunch, or dinner, depending on the time of day. Each recommendation provides detailed nutritional information, helping users select balanced meals. The use case includes options for users to select a specific meal (e.g., breakfast, lunch, or dinner) and then receive detailed nutritional insights about their chosen meal. This feature empowers users to make informed food choices aligned with their dietary goals.

Fitness

NutriLife also integrates fitness tracking capabilities. When the user selects Fitness, they are provided with options for tracking exercises such as bicep curls or squats. By extending the fitness tracking module, the app enables users to start real-time tracking of their repetitions using the device's sensors. The app provides feedback on the user's performance, including the number of reps completed. This feature helps users keep track of their workouts and ensures they are consistently progressing toward their fitness goals.

Statistics

Through the Statistic feature, users can access historical data regarding their dietary habits. The app allows users to select specific dates and view their food records, displaying the associated calorie intake for that day. This feature gives users a detailed view of their past meal choices, enabling them to make better dietary decisions in the future. It offers an includes relationship

CHAPTER 3

with food records and calorie information, helping users monitor their long-term health progress.

3.2.3 Activity Diagram

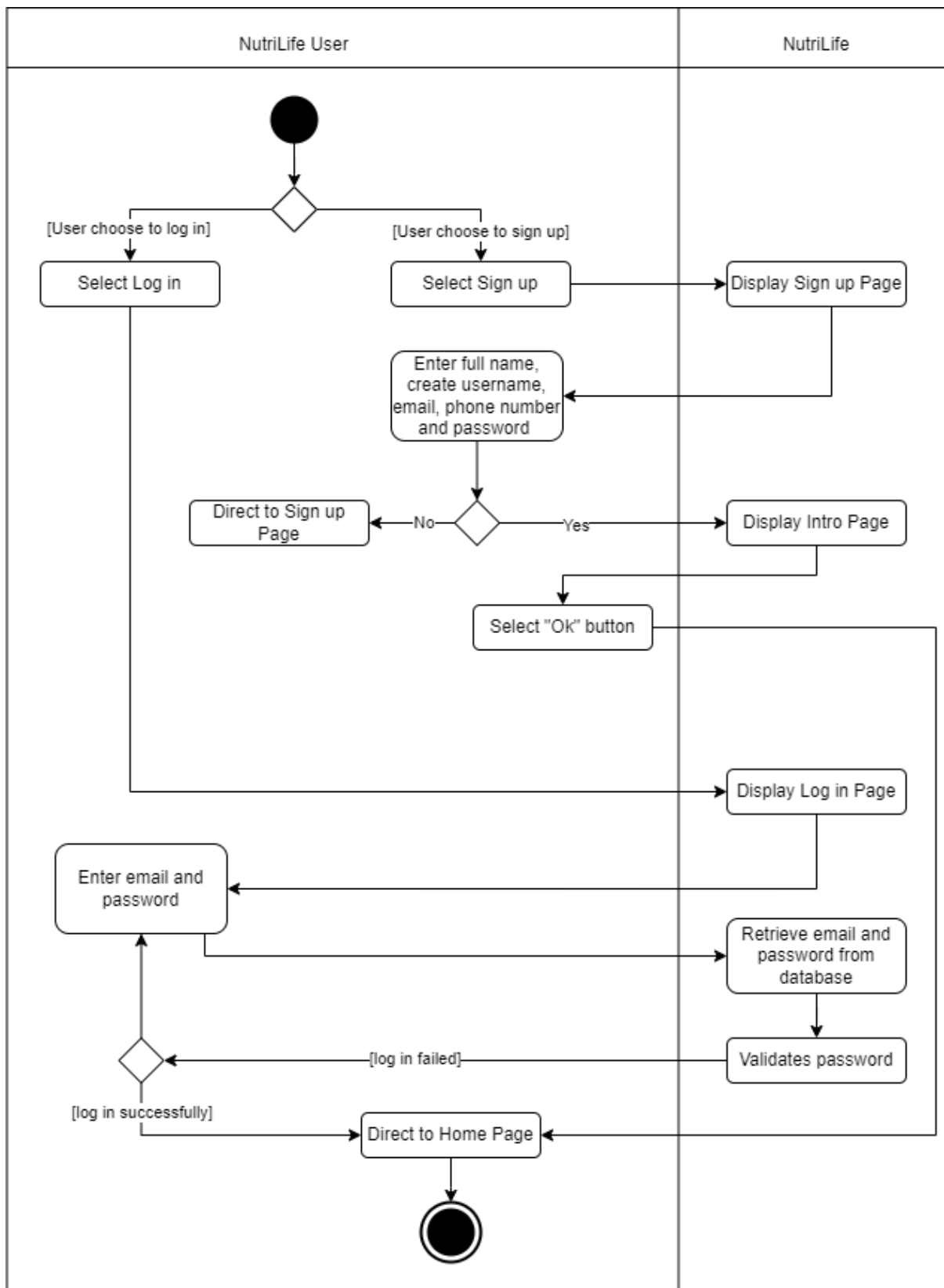


Figure 3.2.3.0: Log in and sign-up activity diagram

The log in and sign-up activity diagram outlines the user authentication process within the NutriLife app. When a new user registers, they fill out a form with personal details like a username, password, and email address. Once the form is submitted, an email verification is sent to confirm the account creation. For existing users, the login process involves entering valid credentials (username and password). If the credentials are incorrect, users can attempt to log in again or use the password recovery feature. Upon successful login, users are directed to the home page, where they can access all personalized features. This process ensures secure access and user data protection.

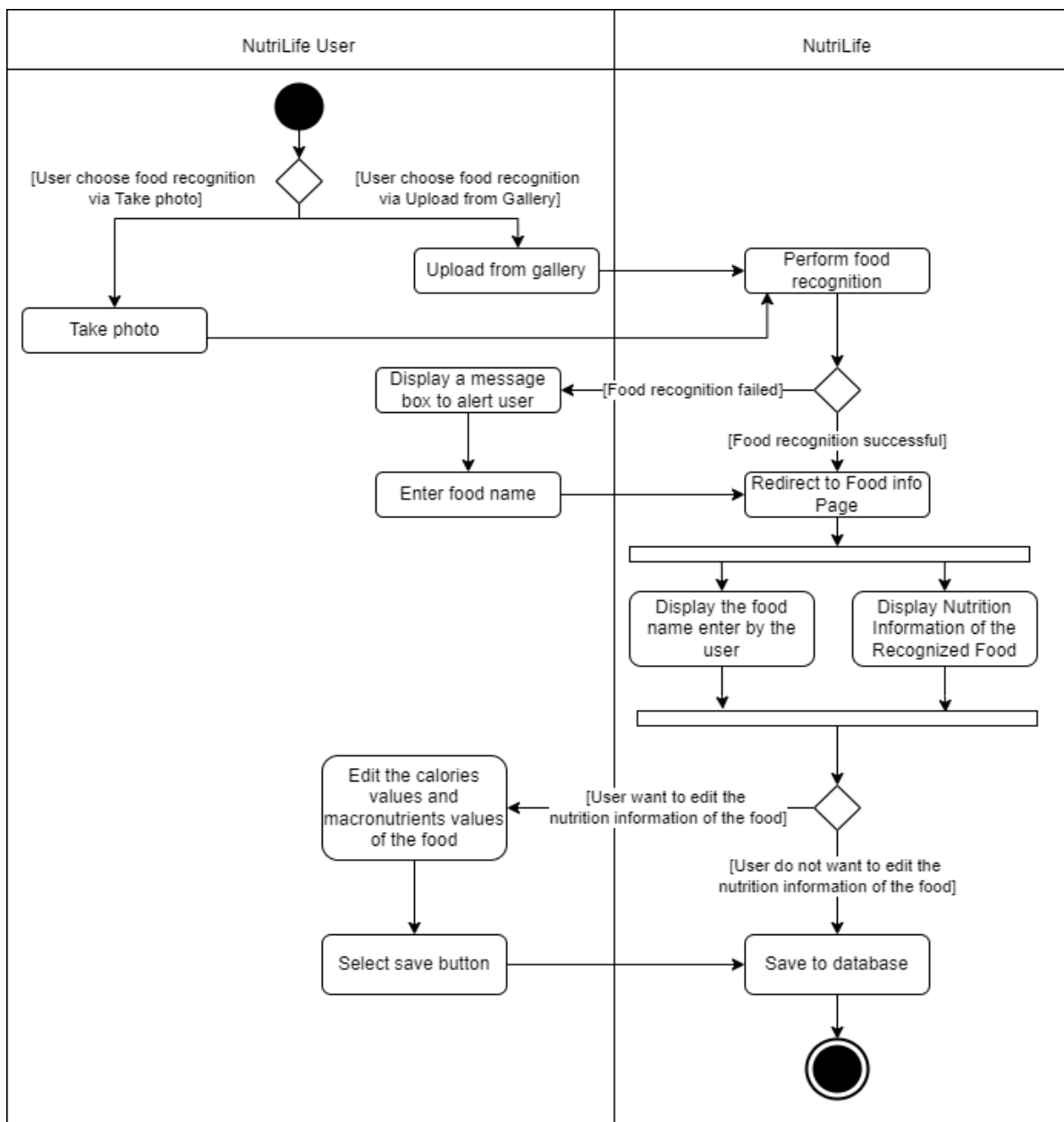


Figure 3.2.3.1: Food recognition activity diagram

The food recognition activity diagram depicts the sequence of actions involved when a user interacts with the food identification system. Users can either take a picture of a meal using their device’s camera or upload an image from the gallery. The image is processed through a Convolutional Neural Network (CNN), which attempts to recognize the food. If the system successfully identifies the food, it displays the associated nutritional information, including calories and macronutrients. In cases where the system is unable to recognize the food, users can manually input the food name to retrieve relevant nutritional data. This module facilitates an easy and interactive way for users to track their food intake.

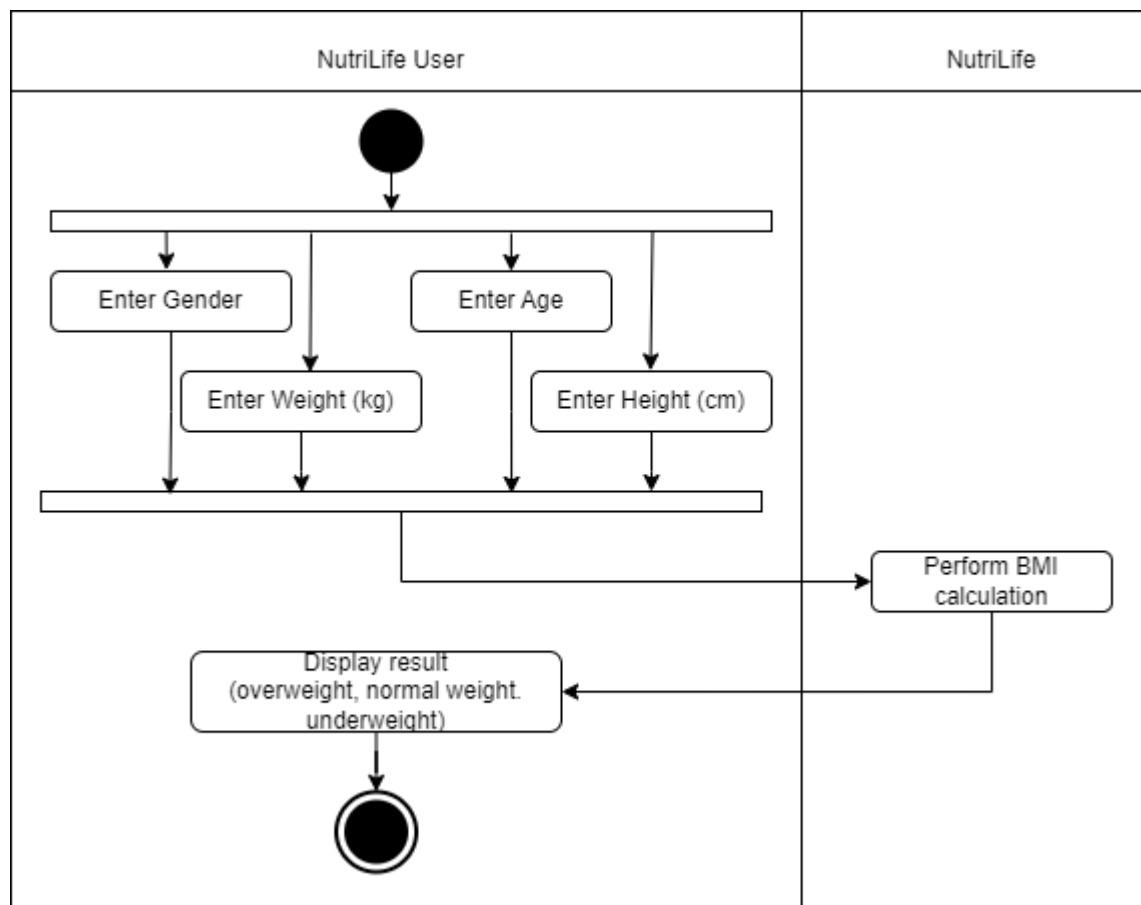


Figure 3.2.3.2: BMI activity diagram

The BMI activity diagram describes the process users follow to calculate their Body Mass Index (BMI). Users input their weight and height into the system, which then calculates the BMI using a standard formula. The calculated BMI is displayed along with a classification indicating whether the user is underweight, normal weight, overweight, or obese. This feature helps users gain insights into their physical health and assists in setting or adjusting fitness goals based on the BMI result.

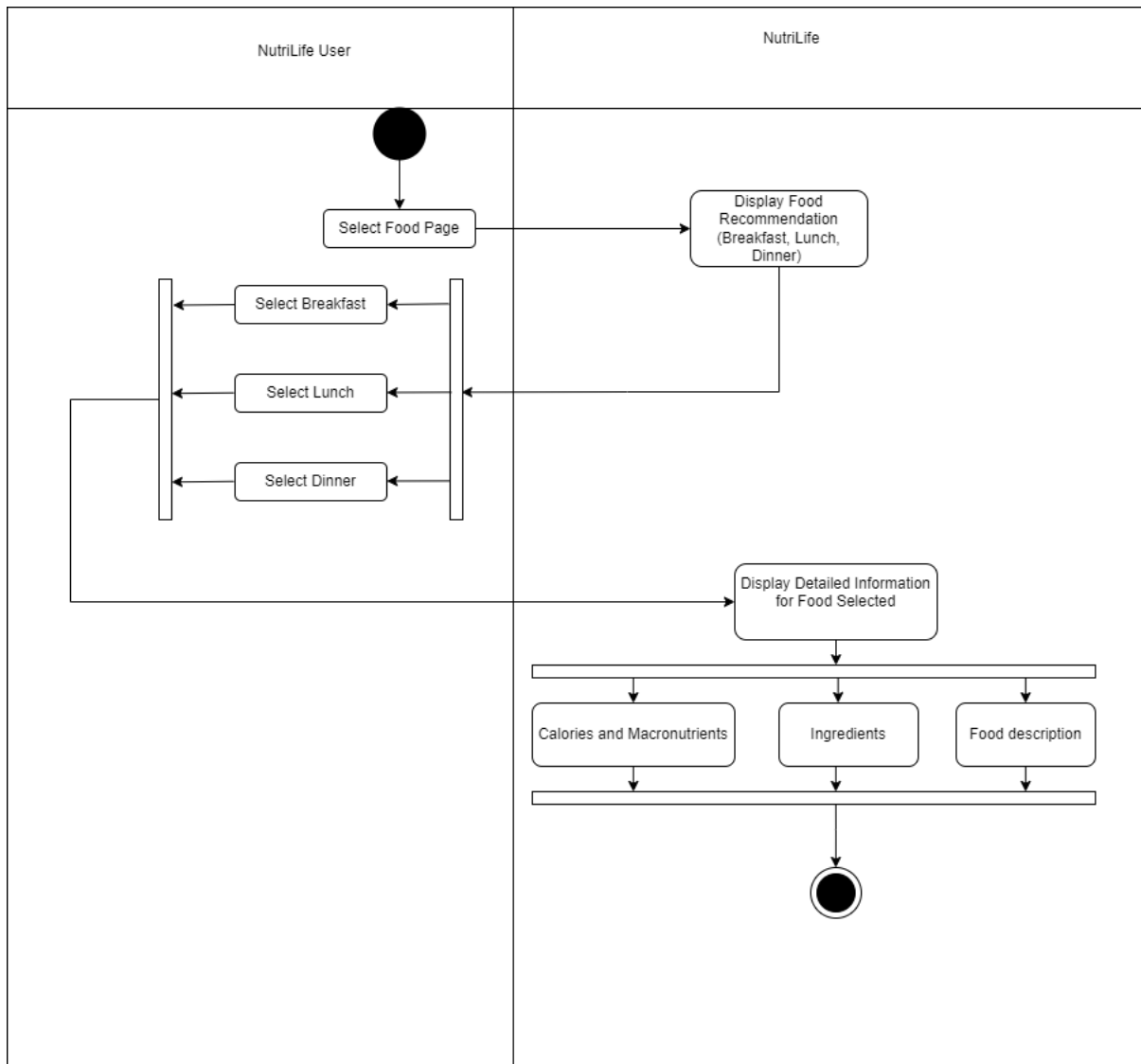


Figure 3.2.3.3: Food page activity diagram

The food page activity diagram describes how users interact with the meal suggestion feature of the NutriLife app. When users access the food page, they are provided with recommendations for simple and easy-to-cook meals based on their dietary preferences and nutritional needs. These suggestions are categorized by meal types, such as breakfast, lunch, or dinner. The page is designed to help users plan their meals quickly and effortlessly, with nutritional information provided for each suggested recipe.

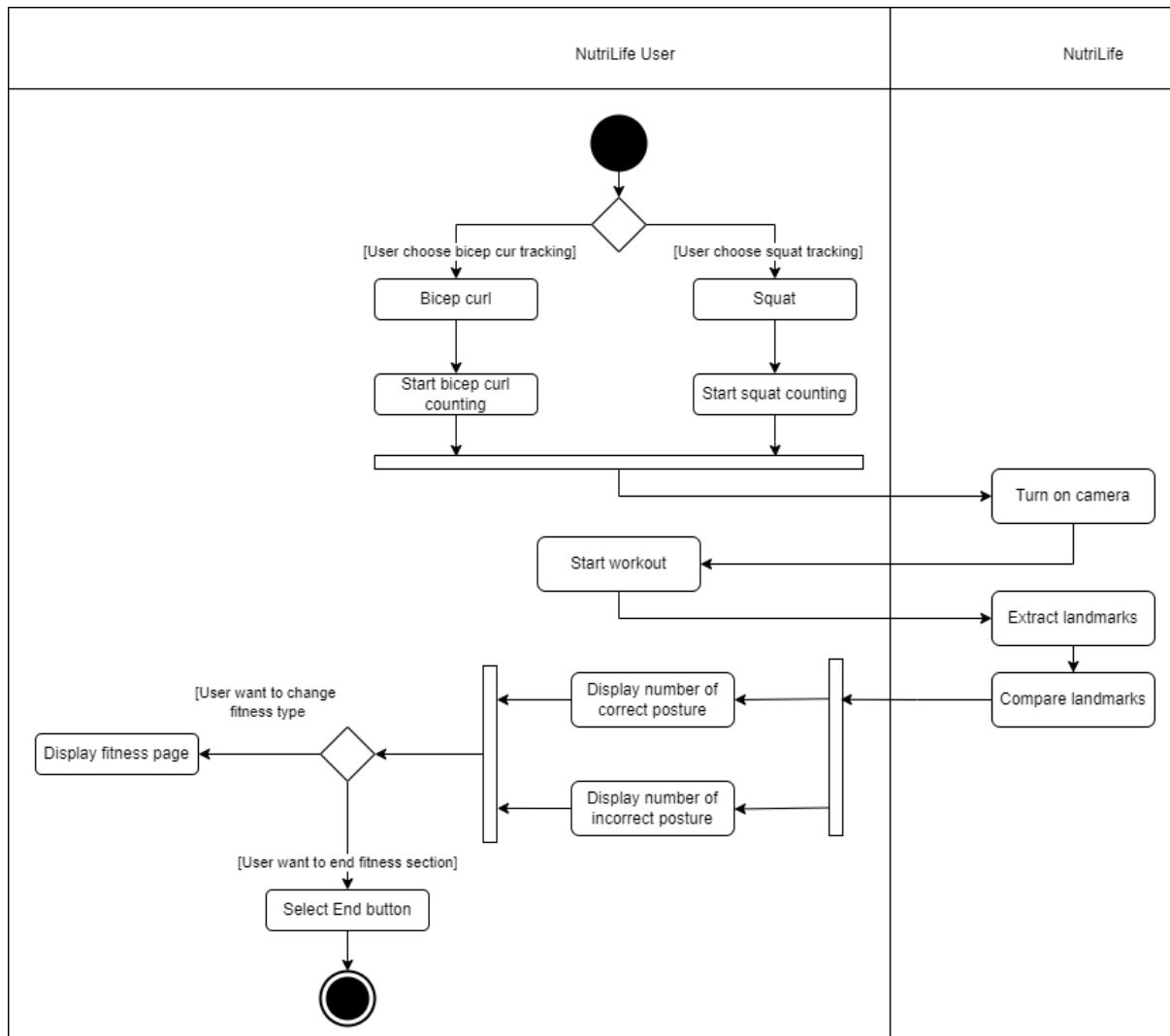


Figure 3.2.3.4: Fitness page activity diagram

The fitness page activity diagram explains how users engage with the fitness tracking system, particularly during exercises like squats and bicep curls. The system uses MediaPipe and machine learning models to analyse the user’s posture in real time, providing instant feedback on form and technique. The system highlights areas that require correction, helping users adjust their posture to avoid injuries and optimize workout efficiency. This feature is crucial for users aiming to improve their exercise technique while ensuring safety during workouts.

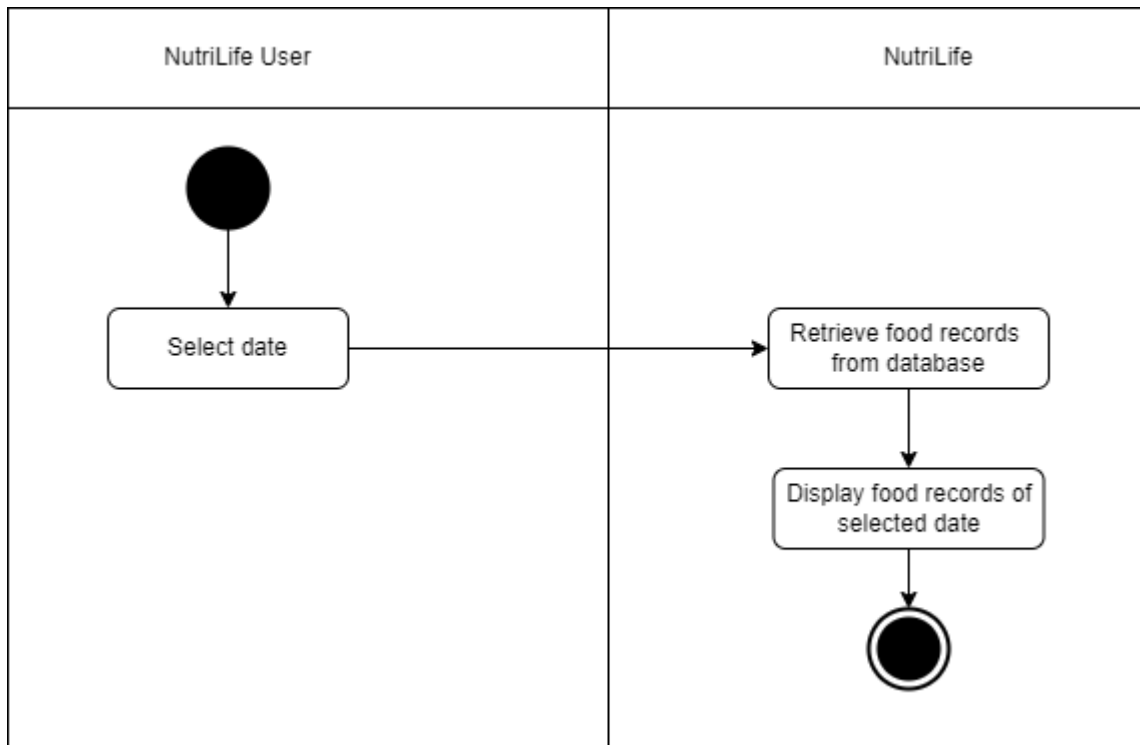


Figure 3.2.3.5:Statistic page activity diagram

The statistics page activity diagram outlines how users can view and interact with their historical food data. Users select a specific date from a calendar view, which allows them to review the meals they consumed on that day, along with detailed nutritional information for each meal. This feature offers users an easy way to track their eating habits over time, helping them make informed decisions about their diet and monitor their long-term progress toward health goals.

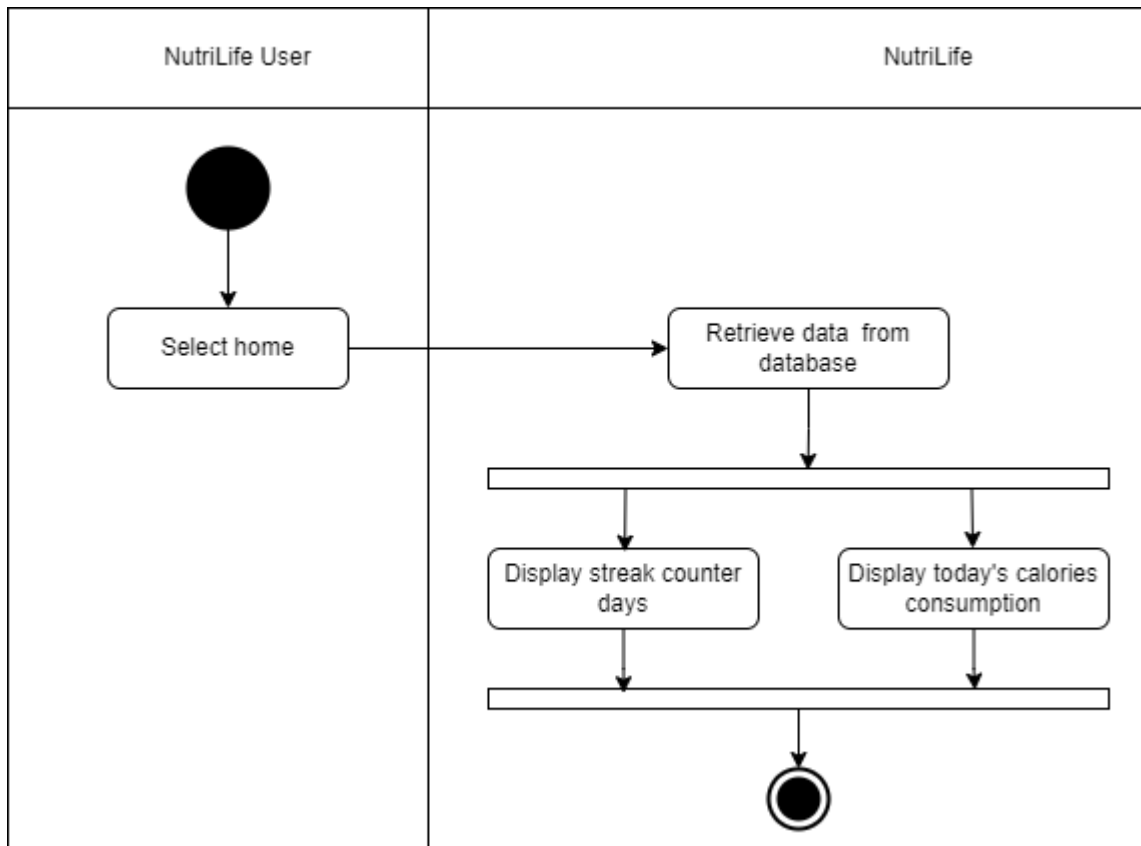


Figure 3.2.3.5:Home page activity diagram

The home page activity diagram highlights how users navigate the core features of the NutriLife app. Upon accessing the home page, users are presented with personalized data such as their daily calorie intake and streak counter, which tracks consecutive days of activity.

3.3 Project Timeline

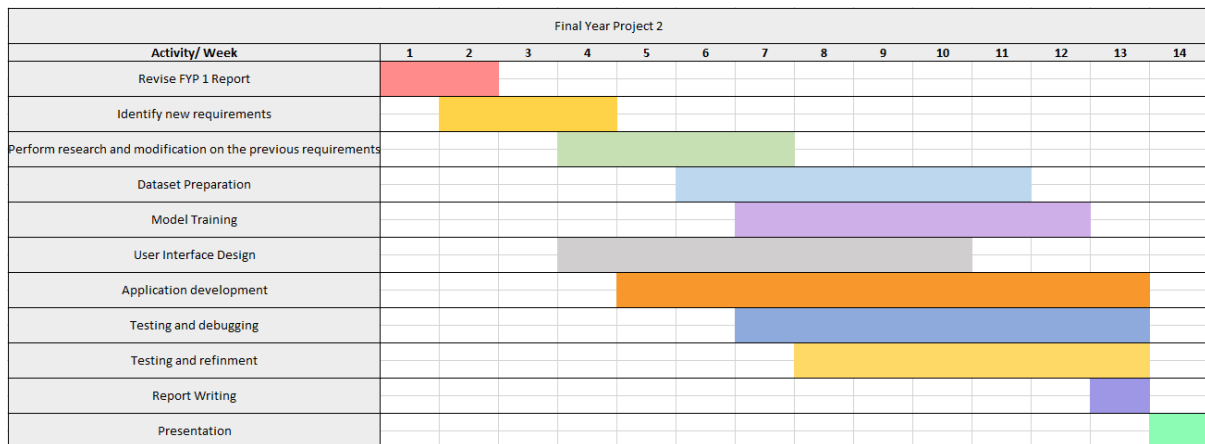


Figure 3.3.0 : Project Timeline for NutriLife

CHAPTER 4 SYSTEM DESIGN

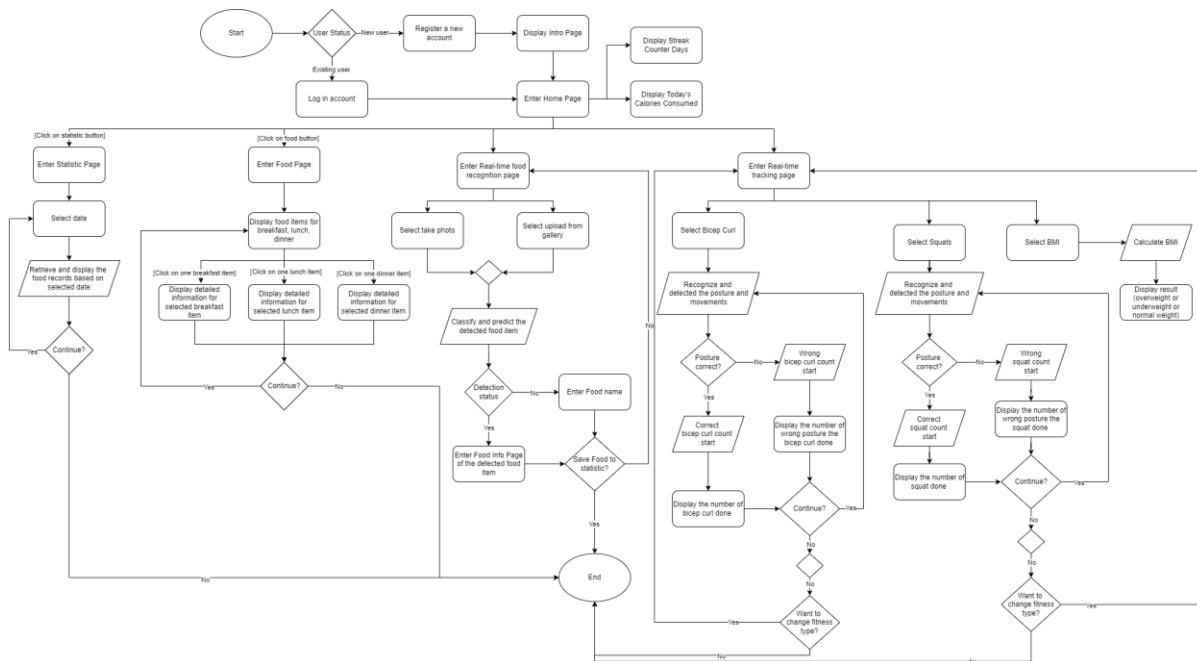


Figure 4.1.0 System flowchart for NutriLife

1. Login/Register New Account Module

This module serves as the user authentication system for the application which allows new users to create an account by providing personal details and setting up credentials, such as a username and password. Existing users can log in using their credentials to access personalized features. The module ensures secure authentication and redirects the user to the home page upon successful registration or login. In case of incorrect login details, the user is prompted to retry. This module is crucial for maintaining personalized user data and offering a tailored experience based on user activities.

2. Home Page Module

This module acts as the central hub of the application, displaying personalized data such as the number of streak counter days (consecutive days of activity) and the calories consumed for the current day. This module provides access to other key features of the system, including the Statistics Page, Real-time Food Recognition, Fitness Tracking, and BMI Calculation modules. The home page dynamically updates based on user activity and serves as the primary point of interaction for navigating to other sections of the application. From here, users can choose

which feature to explore based on their current needs, whether it be tracking their food consumption, engaging in fitness activities, or calculating their BMI.

3. Statistics Page Module

This module enables users to review their past food consumption records, providing a detailed history of their meals. Users can select a specific date from a calendar view to retrieve and display the food items they consumed on that day.

4. Food Recommendation Page module

The page breaks down food records into breakfast, lunch, and dinner, allowing users to click on each meal to view detailed nutritional information. This module offers valuable insights into the user's eating habits over time, aiding in their efforts to maintain a balanced diet. After reviewing the detailed information of each meal, the user can decide whether to continue exploring more records or return to the home page.

6. Real-time Food Recognition Module

This module leverages machine learning to classify and predict food items from images taken by the user's device camera or uploaded from their gallery which allows users to either capture a photo of their meal or select an image from their existing photo library. The system processes the image using a TensorFlow Lite model as mentioned in the model training section in section 3.1.2.1, which attempts to identify the food item. If the system's confidence level is high, it displays the result to the user. In cases where the detection is uncertain or incorrect, the user can manually input the food name. The recognized or manually entered food can then be saved to the user's daily statistics, making it available for future reference in the Statistics Page. This module ensures accurate food logging and contributes to the overall user experience by integrating real-time recognition capabilities.

7. Fitness Tracking Module

This module is designed to monitor and evaluate the user's physical exercises, specifically focusing on bicep curls and squats. Using posture recognition technology, this module detects whether the user is performing the exercise with the correct form. For example, during bicep curls or squats, the system tracks the user's movements and distinguishes between correct and incorrect postures. The system provides real-time feedback on the number of correct and incorrect repetitions. This module motivates users by ensuring they maintain proper form, thus

CHAPTER 4

enhancing the effectiveness of their workouts. After completing a set, the user can choose whether to continue the current exercise, switch to another fitness type, or end the session. The fitness tracking module contributes to the user's overall health by providing accurate performance feedback during exercise routines.

8. BMI Calculation Module

This module allows users to calculate their Body Mass Index (BMI) based on their weight and height inputs. This module provides a simple interface where the user can enter their body metrics, and the system calculates their BMI using the standard formula. Once the BMI is calculated, the result is displayed along with an indication of whether the user falls into the underweight, normal, or overweight category. This module provides essential information regarding the user's body composition and serves as a guideline for managing their health. Users can choose to make additional calculations or return to the home page after viewing their BMI results.

CHAPTER 5 SYSTEM IMPLEMENTATION

5.1 Tools to use

5.1.1 Hardware Specifications

Table 5.1.1 Specifications of laptop

Description	Specifications
Model	VivoBook_ASUSLaptop X409JP_A409JP
Processor	Intel(R) Core (TM) i5-1035G1 CPU
Operating System	Microsoft Windows 11
Graphic	NVIDIA GeForce MX330
Memory	8.00GB RAM
Storage	1TB + 512GB HDD+SDD

Table 5.1.2 Specifications of android device (Oppo Reno 7 5G)

Description	Specifications
Model	CPH2371
Processor	Dimensity 900 Octa-core
Operating System	Android 13
Camera	Front camera: 32MP Rear: 64MP+8MP+2MP
RAM	8.00GB GB + 6.00 GB
Storage	256GB

5.1.2 Software Specifications

The following software is needed to complete the project:

i. Java

Java plays a pivotal role in empowering Android app development. Its installation is indispensable as it stands as one of the officially supported programming languages for Android development. Java provides the fundamental Java runtime environment required for executing Android Studio and efficiently building Android applications.

ii. Android Studio

As the official Integrated Development Environment (IDE) designed specifically for Android app development, Android Studio offers a feature-rich environment equipped with a myriad of tools. Developers benefit from these tools to design captivating user interfaces, write clean and efficient code, rigorously test their applications, and proficiently debug to ensure the highest quality in Android app development.

iii. Python and Google Collab

The integration of Python and Google Collab further enhances the project's capabilities, particularly in training the Convolutional Neural Network (CNN) for the food databases. Python serves as a versatile and widely-used programming language, while Google Collab provides a collaborative environment for executing Python code, particularly suitable for training machine learning models such as CNNs.

iv. Firebase

Complementing the software setup, Firebase by Google is an indispensable inclusion. Renowned for its versatility, Firebase offers a comprehensive mobile and web application development platform, featuring a plethora of services including cloud storage, authentication, real-time database capabilities, and more. These services can seamlessly integrate into the project, enhancing its functionality and features through secure and scalable cloud-based solutions.

v. Jupyter and Mediapipe

NutriLife integrates Jupyter Notebook and Mediapipe to develop, train, and export a fitness tracking model for the NutriLife application. The NutriLife fitness tracking model will focus on recognizing and evaluating fitness activities, such as squats and bicep curls, based on human

CHAPTER 5

pose estimation and posture detection. MediaPipe will serve as the core framework for real-time human pose detection, and Jupyter Notebook will be used for interactive data preprocessing, model training, and exporting the TensorFlow Lite (TFLite) model for deployment in mobile applications.

5.1.3 System Requirements for Users

There are a few fundamental conditions that must be met in order for this application to function properly.

The Android device must meet the following specifications:

i. Operating System:

The minimum API level for the Android device must be Android 7.0 ‘Nougat’ (API level 24) or higher.

ii. Camera Capabilities

The device should have a camera with autofocus and adequate resolution for precise scanning.

iii. Gallery Access

The device must support access to the gallery for uploading and viewing images.

iv. Internet Connectivity

A stable internet connection is required for the application to function properly. This includes fetching user account details, product information, and any additional data required for operation.

The application requires the following permissions:

i. Camera Access

Users must grant permission for the application to access the device's camera for scanning.

ii. Internet Access

Permission to access the internet is necessary for retrieving data.

iii. Gallery Access

Permission to access the device's gallery is needed for uploading and viewing images.

iv. Storage Space

Adequate storage space must be available on the device to install and operate the application. This includes space for caching data, storing scanned food information locally, and managing images from the gallery.

5.1.4 Software Setup (Food Recognition)

Python and TensorFlow Installation

The Python version for this project is 3.10.11 and the TensorFlow version for this project is 2.17.0. With the downloaded installer such as cuDnn, Cuda as shown in Figure 5.1.4.0, TensorFlow then can be used same for Python installer

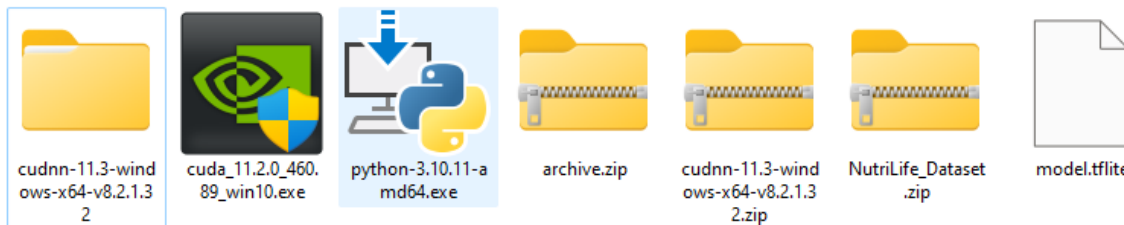


Figure 5.1.4.0 Python and TensorFlow Installation

After downloaded all the required files, download the TensorFlow in Google Collab using the command “pip install tensorflow,” as shown in Figure 5.1.4.1 so that the CNN can be used to train the dataset later.

```
pip install tensorflow
Requirement already satisfied: tensorflow in /usr/local/lib/python3.10/dist-packages (2.15.0)
Requirement already satisfied: numpy<2.0.0, >=1.24.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.26.4)
Requirement already satisfied: google-auth<2.28.0, >=2.27.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.27.1)
Requirement already satisfied: google-auth-oauthlib<0.5.0, >=0.4.6 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.4.6)
Requirement already satisfied: grpcio<1.63.0, >=1.62.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.62.2)
Requirement already satisfied: keras<3.0.0, >=2.10.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.10.0)
Requirement already satisfied: keras-preprocessing>0.4.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.1.3)
Requirement already satisfied: libclang<14.0.0, >=13.0.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (13.0.0)
Requirement already satisfied: ml-dtypes>0.1.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: opt-einsum<3.4.0, >=3.3.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.3.0)
Requirement already satisfied: protobuf<4.25.0, >=4.21.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (4.21.0)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from tensorflow) (69.5.1)
Requirement already satisfied: tensorflow-estimator in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.15.0)
Requirement already satisfied: tensorflow-io-gcs-filesystem>0.14.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.34.0)
Requirement already satisfied: typing-extensions>3.9.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (4.12.0)
Requirement already satisfied: wheel in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.43.0)
Requirement already satisfied: googleapis-common-protos>1.51.0 in /usr/local/lib/python3.10/dist-packages (from google-auth>=2.27.0) (1.63.0)
Requirement already satisfied: cachetools<5.5.0, >=2.0.0 in /usr/local/lib/python3.10/dist-packages (from google-auth>=2.27.0) (5.5.0)
Requirement already satisfied: rsa<4.9.0, >=4.7.0 in /usr/local/lib/python3.10/dist-packages (from google-auth>=2.27.0) (4.9.0)
Requirement already satisfied: pyasn1-modules<0.4.0, >=0.2.8 in /usr/local/lib/python3.10/dist-packages (from google-auth>=2.27.0) (0.4.0)
Requirement already satisfied: requests<3.0.0, >=2.25.1 in /usr/local/lib/python3.10/dist-packages (from google-auth-oauthlib>=0.4.6) (2.32.3)
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packages (from requests>=2.25.1) (2024.7.4)
Requirement already satisfied: idna in /usr/local/lib/python3.10/dist-packages (from requests>=2.25.1) (3.10)
Requirement already satisfied: urllib3 in /usr/local/lib/python3.10/dist-packages (from requests>=2.25.1) (2.2.3)
Requirement already satisfied: charset-normalizer in /usr/local/lib/python3.10/dist-packages (from requests>=2.25.1) (3.3.0)
Requirement already satisfied: tensorflow-io<0.37.0, >=0.37.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow-io-gcs-filesystem>=0.34.0) (0.37.0)
Requirement already satisfied: tensorflow-io-compression<0.4.0, >=0.4.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow-io-gcs-filesystem>=0.34.0) (0.4.0)
```

Figure 5.1.4.1: Install tensorflow in google colab

Dataset Preparation

As shown in Figure 5.1.4.2, eight distinct classes of dataset have prepared in this project which are “half_boiled_egg”, “kaya_toast”, “kuey_teow”, “laksa”, “nasi_lemak”, “popiah”, roti_canai”, “satay”, with a total of 20,000 images.

Name	Date modified	Type	Size
half_boiled_egg	09/04/24 1:21 AM	File folder	
kaya_toast	09/04/24 1:19 AM	File folder	
kuey_teow	09/04/24 1:21 AM	File folder	
laksa	09/04/24 1:21 AM	File folder	
nasi_lemak	09/04/24 1:21 AM	File folder	
popiah	09/04/24 1:22 AM	File folder	
roti_canai	09/04/24 1:22 AM	File folder	
satay	09/04/24 1:21 AM	File folder	

Figure 5.1.4.2: Dataset preparation

Pre requisites and GPU Configuration

```

Pre-requisites

# Import necessary libraries
import tensorflow as tf
import os
import cv2
import imghdr
import zipfile
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from PIL import Image
from sklearn.metrics import confusion_matrix
from google.colab import drive
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Dense, Flatten, Dropout
from tensorflow.keras.preprocessing_image import ImageDataGenerator
from tensorflow.keras.metrics import Precision, Recall, BinaryAccuracy

# Check TensorFlow version and GPU availability
print("Num GPUs Available: ", len(tf.config.list_physical_devices('GPU')))
print("TensorFlow version: ", tf.__version__)

# Check if any GPUs are visible to TensorFlow
device_name = tf.config.list_physical_devices('GPU')
if not device_name:
    print("No GPU found. Proceeding with CPU.")
else:
    print("Found GPU at: {}".format(device_name))

# and also to avoid OOM errors - out of memory; to avoid using all GPU when running large datasets
gpus = tf.config.experimental.list_physical_devices('GPU')
for gpu in gpus:
    tf.config.experimental.set_memory_growth(gpu, True)

Num GPUs Available: 1
TensorFlow version: 2.17.0
Found GPU at: [PhysicalDevice(name='/physical_device:GPU:0', device_type='GPU')]

```

Figure 5.1.4.3: Pre requisites and GPU configuration

After downloading the Tensorflow in the Google Colab, we need to check whether the Tensorflow has connected to the GPU and also need to import necessary libraries to the project as well.

Mount from Google Drive

To manage the large dataset efficiently, the dataset was uploaded as a zip file to Google Drive, and then mounted to Google Colab, instead of uploading the dataset directly to Colab, which is time-consuming. As illustrated in Figure 5.1.4.4, permission to access Google Drive is required. Upon successful extraction of the dataset, the message "Dataset unzipped successfully" will be displayed. In addition, to verify the correct file is being extracted, the subfolders name in the main dataset folder will be printed as well as shown in Figure 5.1.4.5.

```

# Mount Google Drive
drive.mount('/content/drive', force_remount=True)

# Define file path and unzip directory
file_path = '/content/drive/MyDrive/NutriLife/Food2.zip'
unzip_dir = '/content/food_data2'

# Unzip the dataset if it's not already unzipped
if os.path.exists(file_path):
    with zipfile.ZipFile(file_path, 'r') as zip_ref:
        zip_ref.extractall(unzip_dir)
    print("Dataset unzipped successfully!")
else:
    print("Dataset file does not exist.")

Mounted at /content/drive
Dataset unzipped successfully!

```

Figure 5.1.4.4: Mount from google drive

```

Remove corrupted images/ dodgy images

#before start removing the dodgy images, check whether the dataset is correct and really unzipped ald
data_dir = '/content/food_data2'

# list all subfolders
subfolders = [f for f in os.listdir(data_dir) if os.path.isdir(os.path.join(data_dir, f))]

print('Subfolders:')
for folder in subfolders:
    print(folder)

Subfolders:
popiah
nasi_lemak
laksa
roti_canal
kaya_toast
fried_noodles
mixed_rice
fried_rice
fish_and_chips
satay
hamburger

```

Figure 5.1.4.5: To verify correct folder is being extracted

Remove dodgy images or corrupted images in the datasets

```

data_dir = '/content/food_data2'
image_exts = ['.jpg', '.png', '.bmp', '.png']

# Dictionary to keep track of the number of images before and after removal
original_counts = {}
remaining_counts = {}

# Count the original number of images in each subfolder
for image_class in os.listdir(data_dir):
    class_dir = os.path.join(data_dir, image_class)
    if os.path.isdir(class_dir):
        original_count = 0
        for image in os.listdir(class_dir):
            image_path = os.path.join(class_dir, image)
            if any(image.lower().endswith(ext) for ext in image_exts):
                original_count += 1
        original_counts[image_class] = original_count

# Remove corrupted images and count remaining images
total_images_remaining = 0

for image_class in os.listdir(data_dir):
    class_dir = os.path.join(data_dir, image_class)
    if os.path.isdir(class_dir):
        remaining_count = 0
        for image in os.listdir(class_dir):
            image_path = os.path.join(class_dir, image)
            try:
                img = cv2.imread(image_path)
                tip = imgndr.what(image_path)
                if tip not in image_exts:
                    print('Image not in ext list {}'.format(image_path))
                    os.remove(image_path) # Remove the corrupted images
            except:
                remaining_count += 1
                total_images_remaining += 1
            print('Issue with image {}'.format(image_path))
            # os.remove(image_path)

        remaining_counts[image_class] = remaining_count

# Print results
print('Original image counts per class:')
for cls, count in original_counts.items():
    print(f'{cls}: {count} images')

print('\nRemaining image counts per class after removal:')
for cls, count in remaining_counts.items():
    print(f'{cls}: {count} images')

print(f'\nTotal images remaining: {total_images_remaining}')

```

```

Image not in ext list /content/food_data2/fried_noodles/152.jpg
Image not in ext list /content/food_data2/fried_noodles/609.jpg
Image not in ext list /content/food_data2/fried_noodles/531.jpg
Image not in ext list /content/food_data2/fried_noodles/773.jpg
Image not in ext list /content/food_data2/fried_noodles/218.jpg
Image not in ext list /content/food_data2/fried_noodles/700.jpg
Image not in ext list /content/food_data2/fried_noodles/522.jpg
Image not in ext list /content/food_data2/fried_noodles/364.jpg
Image not in ext list /content/food_data2/fried_noodles/644.jpg
Image not in ext list /content/food_data2/fried_noodles/910.jpg
Image not in ext list /content/food_data2/fried_noodles/483.jpg
Image not in ext list /content/food_data2/fried_noodles/914.jpg
Image not in ext list /content/food_data2/fried_noodles/116.jpg
Image not in ext list /content/food_data2/fried_noodles/596.jpg
Image not in ext list /content/food_data2/fried_noodles/754.jpg
Image not in ext list /content/food_data2/fried_noodles/394.jpg
Image not in ext list /content/food_data2/fried_noodles/993.jpg
Image not in ext list /content/food_data2/fried_noodles/395.jpg
Image not in ext list /content/food_data2/mixed_rice/457.jpg

```

```

[0] Image not in ext list /content/food_data2/mixed_rice/877.jpg
Image not in ext list /content/food_data2/mixed_rice/630.jpg
Image not in ext list /content/food_data2/mixed_rice/822.jpg
Image not in ext list /content/food_data2/mixed_rice/113.jpg
Image not in ext list /content/food_data2/mixed_rice/845.jpg
Image not in ext list /content/food_data2/mixed_rice/170.jpg
Image not in ext list /content/food_data2/mixed_rice/189.jpg
Image not in ext list /content/food_data2/satay/990.jpg
Image not in ext list /content/food_data2/satay/323.jpg
Image not in ext list /content/food_data2/satay/39.jpg
Image not in ext list /content/food_data2/satay/310.jpg
Image not in ext list /content/food_data2/satay/35.jpg
Image not in ext list /content/food_data2/satay/164.jpg
Image not in ext list /content/food_data2/satay/111.jpg
Image not in ext list /content/food_data2/satay/487.jpg
Image not in ext list /content/food_data2/satay/985.jpg
Image not in ext list /content/food_data2/satay/187.jpg
Image not in ext list /content/food_data2/satay/640.jpg
Image not in ext list /content/food_data2/satay/482.jpg
Image not in ext list /content/food_data2/satay/997.jpg
Image not in ext list /content/food_data2/satay/181.jpg
Image not in ext list /content/food_data2/satay/51.jpg
Image not in ext list /content/food_data2/satay/32.jpg
Image not in ext list /content/food_data2/satay/409.jpg
Image not in ext list /content/food_data2/satay/974.jpg
Image not in ext list /content/food_data2/satay/183.jpg
Image not in ext list /content/food_data2/satay/967.jpg
Image not in ext list /content/food_data2/satay/270.jpg
Image not in ext list /content/food_data2/satay/254.jpg
Original image counts per Class:
popiah: 1000 Images
nasi_lamak: 1000 Images
laksa: 1000 Images
roti_canis: 1000 Images
kaya_toast: 1000 Images
fried_noodles: 1000 Images
mixed_rice: 1000 Images
fried_rice: 1000 Images
fish_and_chips: 1000 Images
satay: 1000 Images
hamburger: 1000 Images

Remaining image counts per class after removal:

```

Figure 5.1.4.6: Remove dodgy images

Before commencing dataset training, it is essential to remove any corrupted or unreliable images to ensure the quality and reliability of the dataset. This process helps reduce overfitting, minimizes confusion during model training, and enhances the performance of the CNN model. Files that do not match the acceptable extensions (i.e., JPEG, JPG, BMP, PNG) will be

discarded. The initial number of images in each class will be displayed, followed by the total number of remaining images in each class after the removal process as shown in Figure 5.1.4.6.

Split the datasets into testing, training, and validation sets

```

check and verify data

[10] print(data_dir)
print(os.listdir(data_dir))
class_names = os.listdir(data_dir)
print(class_names)

/content/food_data2
['popiah', 'nasi_lenak', 'laksa', 'roti_cantai', 'kaya_toast', 'fried_noodles', 'mixed_rice', 'fried_rice', 'fish_and_chips', 'satay', 'hamburger']
['popiah', 'nasi_lenak', 'laksa', 'roti_cantai', 'kaya_toast', 'fried_noodles', 'mixed_rice', 'fried_rice', 'fish_and_chips', 'satay', 'hamburger']

split the data (test,train,validate)

[11] # import os

# Define paths
base_dir = '/content/food_data2'
split_dir = '/content/food_data_split'
train_dir = os.path.join(split_dir, 'train')
val_dir = os.path.join(split_dir, 'validation')
test_dir = os.path.join(split_dir, 'test')

# Create directories if they don't exist
for directory in [train_dir, val_dir, test_dir]:
    if not os.path.exists(directory):
        os.makedirs(directory)

# Create class subdirectories
for class_name in os.listdir(base_dir):
    class_dir = os.path.join(base_dir, class_name)
    if os.path.isdir(class_dir):
        os.makedirs(os.path.join(directory, class_name), exist_ok=True)

```

Figure 5.1.4.7: Split the datasets

To prepare the dataset for training, it is crucial to split it into three distinct subsets: training, validation, and testing. This approach ensures that the model is trained effectively, tuned for optimal performance, and evaluated accurately.

The training set is the largest portion of the dataset and is used to train the model. By exposing the model to this subset, it can learn and adjust its parameters based on the diverse examples provided. The training set typically constitutes the majority of the dataset, enabling the model to develop a robust understanding of the data.

The validation set serves a different purpose. It is used during the training process to fine-tune hyperparameters and assess the model's performance on unseen data. This subset helps in adjusting model parameters to prevent overfitting, ensuring that the model generalizes well to new, unseen examples. The validation set is crucial for optimizing the model without introducing bias from the training data.

Finally, the testing set is reserved for the final evaluation of the model. This subset is completely separate from the training and validation sets and is used to assess the model's performance in a real-world scenario. By testing the model on this unseen data, we can obtain an unbiased measure of its generalization ability and overall effectiveness.

The dataset is divided into these subsets with a common split ratio of 70% for training, 15% for validation, and 15% for testing. This division helps ensure that the model is trained comprehensively, validated effectively, and tested accurately.

Load data and preparation

```

# Define image size and batch size
img_size = (128, 128)
batch_size = 64

# Data augmentation for training data
train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest'
)

# No augmentation for validation and test data
val_test_datagen = ImageDataGenerator(rescale=1./255)

# Load data
train_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size=img_size,
    batch_size=batch_size,
    class_mode='categorical'
)

validation_generator = val_test_datagen.flow_from_directory(
    val_dir,
    target_size=img_size,
    batch_size=batch_size,
    class_mode='categorical'
)

test_generator = val_test_datagen.flow_from_directory(
    test_dir,
    target_size=img_size,
    batch_size=batch_size,
    class_mode='categorical'
)

```

Found 12253 images belonging to 8 classes.
Found 2624 images belonging to 8 classes.
Found 2633 images belonging to 8 classes.

Figure 5.1.4.8: Data loading and preparation

To effectively train, validate, and test the model, the dataset was loaded and prepared as shown in Figure 5.1.4.8.

Data Augmentation for Training Set

To enhance the training data, the ImageDataGenerator class from TensorFlow's Keras API was utilized. The augmentation parameters applied include:

1. **Rescaling:** Pixel values were normalized to the range [0, 1] by dividing by 255.
2. **Rotation:** Images were randomly rotated within a range of 20 degrees to introduce variability.
3. **Width and Height Shifts:** Images were shifted horizontally and vertically by up to 20% of their dimensions.
4. **Shear:** Random shear transformations up to 20 degrees were applied to simulate perspective changes.
5. **Zoom:** A zoom range of 20% was applied to vary the scale of the images.
6. **Horizontal Flip:** Images were randomly flipped horizontally to increase diversity.
7. **Fill Mode:** Pixels outside the image boundaries were filled using the nearest pixel values.

These augmentations were implemented to improve model generalization by exposing it to a diverse set of transformations.

Preparation for Validation and Testing Sets

For the validation and testing datasets, no augmentation was applied to preserve the authenticity of the data. Only rescaling was performed to normalize pixel values to the range [0, 1]. This approach ensures that the evaluation data accurately represents real-world scenarios.

Loading Data

The `flow_from_directory` method from `ImageDataGenerator` was used to load the images from directories into the training, validation, and testing generators. The following configurations were applied:

1. **Training Data:** Images were loaded from the `train_dir`, resized to 128x128 pixels, and processed in batches of 64. The class mode was set to 'categorical' to handle multiple classes.
2. **Validation Data:** Images were loaded from the `val_dir`, resized to 128x128 pixels, and processed in batches of 64. The class mode was also set to 'categorical'.
3. **Testing Data:** Images were loaded from the `test_dir`, resized to 128x128 pixels, and processed in batches of 64. The class mode remained 'categorical'.

To confirm successful data loading, the number of samples in each dataset was printed. This step ensured that the data was correctly processed and that the appropriate number of images were available for training, validation, and testing

Build CNN

▼ Build CNN

```

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout

model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(128, 128, 3)),
    MaxPooling2D((2, 2)),
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Flatten(),
    Dense(512, activation='relu'),
    Dropout(0.5),
    Dense(8, activation='softmax') # num of classes in my subfolders as shown above
])

```

Figure 5.1.4.9: Build the CNN layer

To develop an effective image classification model for the NutriLife, a Convolutional Neural Network (CNN) was designed.

CHAPTER 5

The model begins with a Convolutional Layer featuring 32 filters, each with a 3x3 kernel size. This layer uses the ReLU activation function to introduce non-linearity and process the input images of size 128x128 with 3 color channels (RGB). The feature maps produced by this layer are then downsampled by a Max-Pooling Layer with a pool size of 2x2, which reduces the spatial dimensions and helps in capturing the most prominent features.

Following this, a second Convolutional Layer with 64 filters of the same kernel size is applied, again using ReLU activation. This is followed by another Max-Pooling Layer to further downsample the feature maps. A third Convolutional Layer with 128 filters is added, enhancing the model's ability to learn complex features. A final Max-Pooling Layer is used to reduce the spatial dimensions of the feature maps from this layer.

The output of the last convolutional layer is then flattened using a Flattening Layer, which transforms the 2D feature maps into a 1D vector. This flattened output is fed into a Fully Connected Layer with 512 units and ReLU activation. This layer helps the model learn high-level representations of the features extracted by the convolutional layers. To mitigate overfitting, a Dropout Layer with a dropout rate of 0.5 is included, randomly setting 50% of the neurons to zero during training.

Finally, the model concludes with a Dense Output Layer consisting of 8 units and using the softmax activation function. This layer produces the probability distribution over the 8 classes, corresponding to the number of classes in the classification task.

Train model

```

from PIL import Image

def convert_image_to_rgba(image_path):
    image = Image.open(image_path)
    if image.mode in ("P", "LA") or (image.mode == "L" and "transparency" in image.info):
        image = image.convert("RGBA")
    return image

```

Figure 5.1.4.10: Convert the image into RGBA model

The `convert_image_to_rgba` function converts an image to RGBA mode (Red, Green, Blue, Alpha) to ensure it includes an alpha channel for transparency. It handles images in palette-based ("P"), grayscale with alpha ("LA"), or grayscale with transparency ("L") modes by converting them to RGBA if necessary. This ensures uniform image format compatibility.

```

# Compile the model
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])

# Convert Keras generators to TensorFlow datasets
train_dataset = tf.data.Dataset.from_generator(
    lambda: train_generator,
    output_signature=(
        tf.TensorSpec(shape=(None, 128, 128, 3), dtype=tf.float32),
        tf.TensorSpec(shape=(None, len(train_generator.class_indices)), dtype=tf.float32)
    )
)

validation_dataset = tf.data.Dataset.from_generator(
    lambda: validation_generator,
    output_signature=(
        tf.TensorSpec(shape=(None, 128, 128, 3), dtype=tf.float32),
        tf.TensorSpec(shape=(None, len(validation_generator.class_indices)), dtype=tf.float32)
    )
)

# Apply .repeat() to the datasets to ensure they loop indefinitely
train_dataset = train_dataset.repeat() # No need to batch again, as generators already provide batches
validation_dataset = validation_dataset.repeat()

# Train the model
history = model.fit(
    train_dataset,
    steps_per_epoch=len(train_generator),
    epochs=30,
    validation_data=validation_dataset,
    validation_steps=len(validation_generator)
)

```

Figure 5.1.4.11: Compiling and training the model

Next, to train the CNN model, the following steps were carried out as shown in figure 5.1.4.11.

1. Model Compilation

The model was compiled using the Adam optimizer, which dynamically adjusts the learning rate to improve training efficiency. The loss function selected was `categorical_crossentropy`, suitable for handling multi-class classification problems. Accuracy was used as the performance metric to evaluate how well the model is classifying the images during training.

2. Conversion to TensorFlow Datasets

The data from Keras generators for training and validation were converted into TensorFlow datasets. This conversion was done to facilitate efficient data handling and pre-processing. The

datasets were set up to output batches of images and their corresponding labels, structured to match the requirements of the CNN model.

3. Dataset Repeating

To ensure uninterrupted data flow during model training, the datasets were configured to repeat indefinitely. This is crucial as it allows the model to continuously receive data for training without needing to restart the dataset manually.

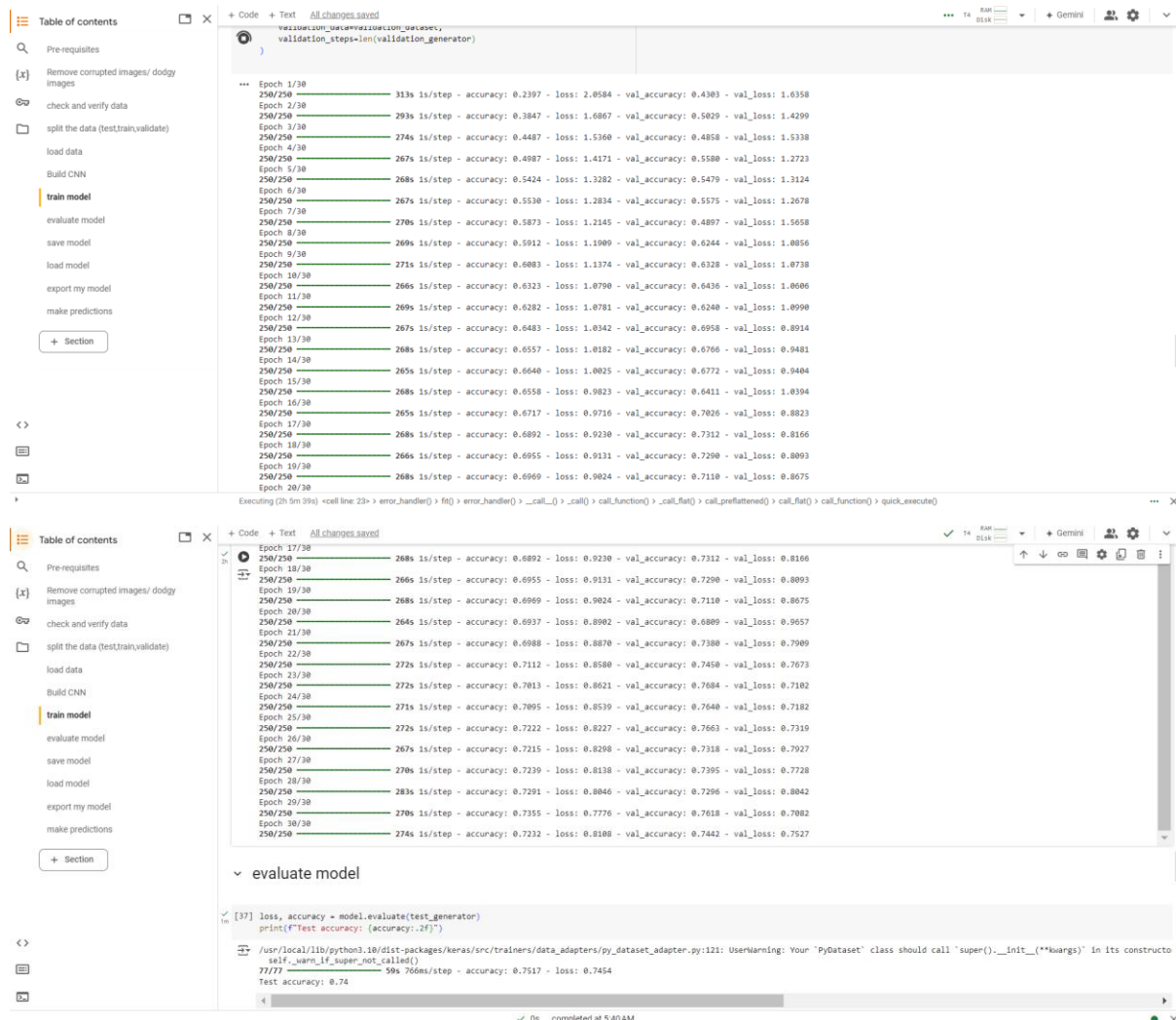


Figure 5.1.4.12: Running epochs

4. Model Training

The model was trained over 30 epochs as shown in figure 5.1.4.12, using the converted TensorFlow datasets for both training and validation. The number of steps per epoch and validation steps were based on the number of batches provided by the data generators. This setup ensures that the model is exposed to the full dataset multiple times, improving its ability to learn and generalize from the data.

Validating the accuracy



```
[37] loss, accuracy = model.evaluate(test_generator)
print(f"Test accuracy: {accuracy:.2f}")

/usr/local/lib/python3.10/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121: UserWarning: Your 'PyDataset' class should call 'super().__init__(**kwargs)' in its constructor
self._warn_if_super_not_called()
77777 ----- 99% 766ms/step - accuracy: 0.7517 - loss: 0.7454
Test accuracy: 0.74
```

Figure 5.1.4.13: Accuracy of 74% for the model

As shown in the figure above, the model achieved an accuracy of 74% on the test dataset. This performance metric indicates how well the model generalizes to new, unseen data after training.



```
print(f"Test accuracy: {accuracy:.2f}")

import matplotlib.pyplot as plt

plt.figure(figsize=(12, 4))

# Plot training & validation accuracy values
plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'])

# Plot training & validation loss values
plt.subplot(1, 2, 2)
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend(['Train', 'Validation'])

plt.show()
```

Figure 5.1.4.14: Illustrate the model accuracy and model loss graph

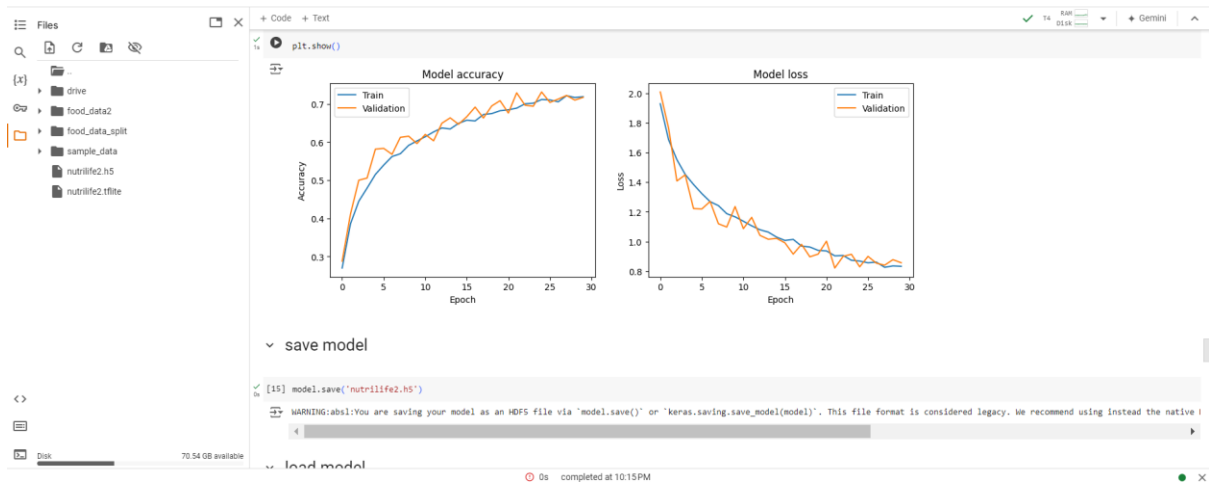


Figure 5.1.4.15: Graph of model accuracy and model loss

The figure 5.1.4.15 illustrates the impact of various factors such as the number of datasets, dense layers, and batch size on the model's accuracy. To ensure robust results and mitigate issues such as overfitting or underfitting, each factor was tested at least 10 times as shown in section 3.1.2.1. In the graphical representation:

- The **blue line** represents the accuracy of the training datasets.
- The **orange line** represents the accuracy of the validation datasets.

- **Model Accuracy:** This plot shows the accuracy of both the training and validation datasets across epochs. It helps in assessing how well the model is learning and generalizing.
- **Model Loss:** This plot illustrates the loss values for both the training and validation datasets. It provides insights into how the model's error decreases over time, reflecting its learning process.

Confusion matrix

To evaluate the model's classification performance in detail, a confusion matrix was generated. This analysis provides insights into the model's accuracy across different classes and highlights any areas of confusion.

```

import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# Define class names again (in the same order as your model was trained)
class_names = ['popiah', 'nasi_lematik', 'kuey_teow', 'laksa', 'roti_canai', 'half_boiled_egg', 'kaya_toast', 'satay']

# Use your existing test generator
test_generator = val_test_datagen.flow_from_directory(
    test_dir,
    target_size=(128, 128),
    batch_size=20,
    class_mode='categorical',
    shuffle=False
)

# Predict the classes for the test set
predictions = model.predict(test_generator, steps=test_generator.samples // test_generator.batch_size + 1)
predicted_classes = np.argmax(predictions, axis=1)

# Get the true labels from the generator
true_classes = test_generator.classes

# Generate the confusion matrix
cm = confusion_matrix(true_classes, predicted_classes)

# Display the confusion matrix
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=class_names)
disp.plot(cmap=plt.cm.Blues)
plt.show()
    
```

Figure 5.1.4.16: Plotting the confusion matrix

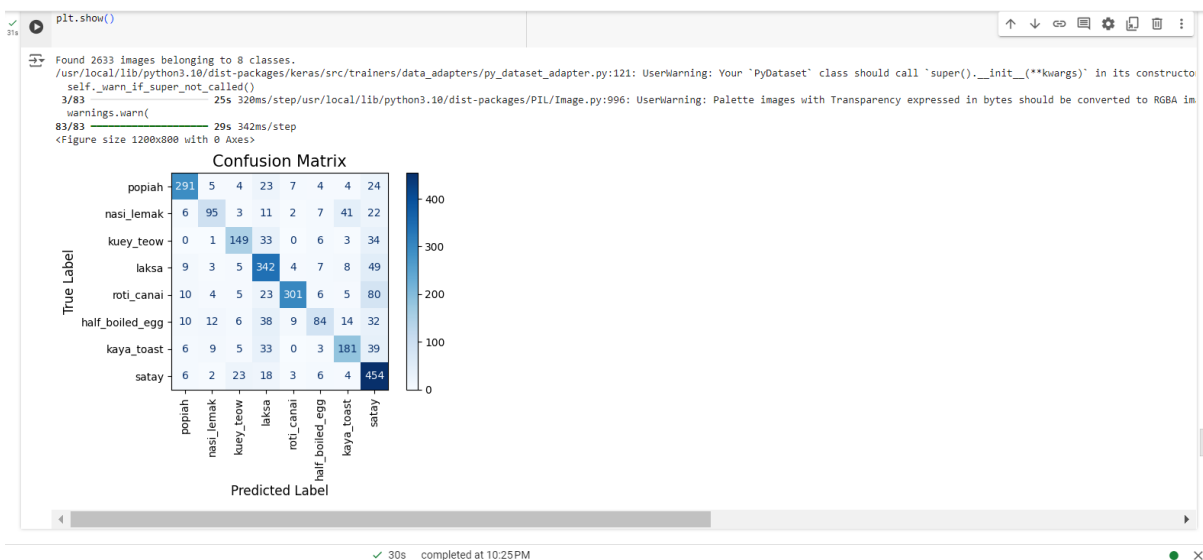


Figure 5.1.4.17: The confusion matrix

Saving the model

▼ save model

```
[ ] model.save('nutrillife2.h5')
```

▼ load model

```
[ ] from tensorflow.keras.models import load_model  
  
model = load_model('nutrillife2.h5')
```

▼ export my model

```
[ ] # Convert the model to TensorFlow Lite format  
converter = tf.lite.TFLiteConverter.from_keras_model(model)  
tflite_model = converter.convert()  
  
# Save the TFLite model to a file  
with open('nutrillife2.tflite', 'wb') as f:  
    f.write(tflite_model)
```

Figure 5.1.4.18: Save and export model in TensorFlow Lite format

To integrate our machine learning model into an Android app, we use a converter to transform the model into a format suitable for mobile devices. Specifically, the model is converted to a .tflite (TensorFlow Lite) format. This process ensures that the model can run efficiently on Android devices.

5.1.5 Software Setup (Fitness tracking - Squat)

```

In [1]: 1 import cv2
2 import mediapipe as mp
3 import numpy as np
4 import pandas as pd
5
6 mp_pose = mp.solutions.pose
7 pose = mp_pose.Pose()
8 mp_drawing = mp.solutions.drawing_utils
9
10 cap = cv2.VideoCapture(0)
11 data = []
12
13 def calculate_angle(a, b, c):
14     a = np.array(a)
15     b = np.array(b)
16     c = np.array(c)
17
18     radians = np.arctan2(c[1] - b[1], c[0] - b[0]) - np.arctan2(a[1] - b[1], a[0] - b[0])
19     angle = np.abs(radians * 180.0 / np.pi)
20
21     if angle > 180.0:
22         angle = 360 - angle
23
24     return angle
25
26 while cap.isOpened():
27     ret, frame = cap.read()
28
29     image = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
30     results = pose.process(image)
31     image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
32
33     try:
34         landmarks = results.pose_landmarks.landmark
35
36         # Get key point coordinates for squats
37         hip = [landmarks[mp_pose.PoseLandmark.LEFT_HIP.value].x, landmarks[mp_pose.PoseLandmark.LEFT_HIP.value].y]
38         knee = [landmarks[mp_pose.PoseLandmark.LEFT_KNEE.value].x, landmarks[mp_pose.PoseLandmark.LEFT_KNEE.value].y]
39         ankle = [landmarks[mp_pose.PoseLandmark.LEFT_ANKLE.value].x, landmarks[mp_pose.PoseLandmark.LEFT_ANKLE.value].y]
40
41         # Calculate the angle
42         knee_angle = calculate_angle(hip, knee, ankle)
43
44         # Label the stage of the squat
45         stage = "up" if knee_angle > 160 else "down" if knee_angle < 90 else None
46
47         if stage:
48             data.append(hip + knee + ankle + [stage])
49
50     except Exception as e:
51         print(e)
52
53     mp_drawing.draw_landmarks(image, results.pose_landmarks, mp_pose.POSE_CONNECTIONS)
54     cv2.imshow("Squat Data Collection", image)
55
56     if cv2.waitKey(10) & 0xFF == ord('q'):
57         break
58
59 cap.release()
60 cv2.destroyAllWindows()
61
62 # Save the collected data to CSV
63 df = pd.DataFrame(data, columns=['hip_x', 'hip_y', 'knee_x', 'knee_y', 'ankle_x', 'ankle_y', 'stage'])
64 df.to_csv('squat_data.csv', index=False)
65

```

Figure 5.1.5.0: Data collection and model training for squat

To implement a system for tracking and analyzing squat exercises, a critical first step is to develop and train a machine learning model capable of classifying the stages of a squat. This process begins with preparing the dataset as shown in the section 3.1.2.2. The dataset is then processed to extract these coordinates as features and to label each squat instance as either "up" or "down."

Label encoding is used to convert these categorical labels into numerical values, specifically 0 for "up" and 1 for "down." This encoding is essential for the model's training process.

Additionally, the input features are normalized using a StandardScaler, which adjusts the range of the data to improve the model's performance and convergence.

The next step involves splitting the dataset into training and testing sets, with 80% of the data allocated for training and 20% reserved for testing. This split is crucial for evaluating the model's performance on unseen data and ensuring its generalizability.

The model itself is a neural network with a defined architecture consisting of an input layer with 64 units and ReLU activation, a hidden layer with 32 units and ReLU activation, and an output layer with 2 units and softmax activation to classify the squat stages. The model is compiled using the Adam optimizer and sparse categorical crossentropy loss function and is trained for 10 epochs with a 10% validation split to monitor its performance throughout the training process.

After training, the model is evaluated on the test set to determine its accuracy. The resulting accuracy metric provides insight into how well the model performs in classifying squat stages. Finally, the trained model is saved to a file named `squat_model.h5`, making it available for future use in the tracking system.

```
In [2]: 1 import tensorflow as tf
2 import pandas as pd
3 from sklearn.model_selection import train_test_split
4 from sklearn.preprocessing import LabelEncoder
5 from sklearn.preprocessing import StandardScaler
6
7 # Load the dataset
8 data = pd.read_csv('squat_data.csv')
9
10 # Prepare the data
11 X = data[['hip_x', 'hip_y', 'knee_x', 'knee_y', 'ankle_x', 'ankle_y']].values
12 y = data['stage'].values
13
14 # Encode the Labels (up = 0, down = 1)
15 label_encoder = LabelEncoder()
16 y_encoded = label_encoder.fit_transform(y)
17
18 # Normalize the input features
19 scaler = StandardScaler()
20 X_scaled = scaler.fit_transform(X)
21
22 # Split the data into training and test sets
23 X_train, X_test, y_train, y_test = train_test_split(X_scaled, y_encoded, test_size=0.2, random_state=42)
24
25 # Define the model architecture
26 model = tf.keras.Sequential([
27     tf.keras.layers.Dense(64, activation='relu', input_shape=(X_train.shape[1],)),
28     tf.keras.layers.Dense(32, activation='relu'),
29     tf.keras.layers.Dense(2, activation='softmax') # 2 output classes: "up" and "down"
30 ])
31
32 # Compile the model
33 model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
34
35 # Train the model
36 model.fit(X_train, y_train, epochs=10, validation_split=0.1)
37
38 # Evaluate the model
39 loss, accuracy = model.evaluate(X_test, y_test)
40 print(f'Test accuracy: {accuracy}')
41
42 # Save the trained model
43 model.save('squat_model.h5')
44
```

Figure 5.1.5.1(a): Conversion of Trained Model to TensorFlow Lite Format

```

Epoch 1/10
111/111 ----- 2s 3ms/step - accuracy: 0.9135 - loss: 0.2890 - val_accuracy: 0.9771 - val_loss: 0.1214
Epoch 2/10
111/111 ----- 0s 2ms/step - accuracy: 0.9915 - loss: 0.0556 - val_accuracy: 0.9771 - val_loss: 0.1134
Epoch 3/10
111/111 ----- 0s 2ms/step - accuracy: 0.9906 - loss: 0.0499 - val_accuracy: 0.9771 - val_loss: 0.0991
Epoch 4/10
111/111 ----- 0s 2ms/step - accuracy: 0.9889 - loss: 0.0464 - val_accuracy: 0.9771 - val_loss: 0.0750
Epoch 5/10
111/111 ----- 0s 1ms/step - accuracy: 0.9891 - loss: 0.0310 - val_accuracy: 0.9771 - val_loss: 0.0546
Epoch 6/10
111/111 ----- 0s 1ms/step - accuracy: 0.9974 - loss: 0.0169 - val_accuracy: 0.9898 - val_loss: 0.0347
Epoch 7/10
111/111 ----- 0s 1ms/step - accuracy: 0.9977 - loss: 0.0110 - val_accuracy: 0.9898 - val_loss: 0.0222
Epoch 8/10
111/111 ----- 0s 2ms/step - accuracy: 0.9993 - loss: 0.0063 - val_accuracy: 0.9898 - val_loss: 0.0181
Epoch 9/10
111/111 ----- 0s 2ms/step - accuracy: 0.9984 - loss: 0.0064 - val_accuracy: 0.9898 - val_loss: 0.0166
Epoch 10/10
111/111 ----- 0s 1ms/step - accuracy: 0.9988 - loss: 0.0048 - val_accuracy: 0.9898 - val_loss: 0.0154
31/31 ----- 0s 2ms/step - accuracy: 0.9984 - loss: 0.0068

```

Figure 5.1.5.1(b): Conversion of Trained Model to TensorFlow Lite Format

To integrate the squat classification model into an Android application, it is necessary to convert the trained TensorFlow model into TensorFlow Lite format. TensorFlow Lite (TFLite) is optimized for mobile and embedded devices, enabling efficient model deployment and inference on smartphones.

First, the pre-trained model, saved as `squat_model.h5`, is loaded using TensorFlow. This model was previously trained to classify squat stages and is now ready for conversion. Using TensorFlow's `TFLiteConverter`, the Keras model is converted into TFLite format. This conversion is crucial as it reduces the model's size and optimizes it for mobile devices, ensuring that it runs efficiently and quickly on Android devices.

The converted TFLite model is then saved to a file named `squat_model.tflite`. This file is compact and optimized for performance on mobile devices, making it suitable for deployment within an Android application.

By converting the model to TensorFlow Lite format, the squat classification model can now be integrated into mobile applications, enabling real-time analysis and feedback on squat exercises directly from a user's smartphone.

```
In [3]: 1 import tensorflow as tf
2
3 # Load the trained model
4 model = tf.keras.models.load_model('squat_model.h5')
5
6 # Convert the model to TensorFlow Lite format
7 converter = tf.lite.TFLiteConverter.from_keras_model(model)
8 tflite_model = converter.convert()
9
10 # Save the TFLite model
11 with open('squat_model.tflite', 'wb') as f:|
12     f.write(tflite_model)
13
```

Figure 5.1.5.2: Save model

The model is converted to a .tflite (TensorFlow Lite) format. This process ensures that the model can run efficiently on Android devices.

5.1.6 Software Setup (Fitness tracking – Bicep curl)

```

In [1]: 1 import cv2
        2 import mediapipe as mp
        3 import numpy as np
        4 import pandas as pd
        5
        6 mp_pose = mp.solutions.pose
        7 pose = mp_pose.Pose()
        8 mp_drawing = mp.solutions.drawing_utils
        9
        10 cap = cv2.VideoCapture(0)
        11 data = []
        12
        13 def calculate_angle(a, b, c):
        14     a = np.array(a)
        15     b = np.array(b)
        16     c = np.array(c)
        17
        18     radians = np.arctan2(c[1] - b[1], c[0] - b[0]) - np.arctan2(a[1] - b[1], a[0] - b[0])
        19     angle = np.abs(radians * 180.0 / np.pi)
        20
        21     if angle > 180.0:
        22         angle = 360 - angle
        23
        24     return angle
        25
        26 while cap.isOpened():
        27     ret, frame = cap.read()
        28
        29     image = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        30     results = pose.process(image)
        31     image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
        32
        33     try:
        34         landmarks = results.pose_landmarks.landmark
        35         shoulder = [landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value].x, landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value].y]
        36         elbow = [landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value].x, landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value].y]
        37         wrist = [landmarks[mp_pose.PoseLandmark.LEFT_WRIST.value].x, landmarks[mp_pose.PoseLandmark.LEFT_WRIST.value].y]
        38
        39         elbow_angle = calculate_angle(shoulder, elbow, wrist)
        40         stage = "down" if elbow_angle > 100 else "up" if elbow_angle < 50 else None
        41
        42         if stage:
        43             data.append(shoulder + elbow + wrist + [stage])
        44
        45     except Exception as e:
        46         print(e)
        47
        48     mp_drawing.draw_landmarks(image, results.pose_landmarks, mp_pose.POSE_CONNECTIONS)
        49     cv2.imshow('Pose Data Collection', image)
        50
        51     if cv2.waitKey(10) & 0xFF == ord('q'):
        52         break
        53
        54 cap.release()
        55 cv2.destroyAllWindows()
        56
        57 # Save collected data
        58 df = pd.DataFrame(data, columns=['shoulder_x', 'shoulder_y', 'elbow_x', 'elbow_y', 'wrist_x', 'wrist_y', 'stage'])
        59 df.to_csv('bicep_curl_data.csv', index=False)
        60

```

Figure 5.1.6.0: Data collection and model training for bicep curl

To collect data for classifying bicep curl exercises, we use a combination of computer vision and pose estimation techniques. The process begins by initializing the MediaPipe Pose module, which is used for detecting body landmarks, and the drawing utilities for visualizing these landmarks on video frames. We then start video capture from a webcam using OpenCV to access the real-time video feed.

During the data collection process, each frame captured from the webcam is converted to RGB format and processed by MediaPipe Pose to extract landmark coordinates. The key landmarks for this exercise are the shoulder, elbow, and wrist. Using these coordinates, we calculate the angle between the shoulder, elbow, and wrist to determine the stage of the bicep

curl exercise. The angle is used to classify the exercise stage as either "down" when the arm is extended or "up" when the arm is bent.

The collected data as shown in section 3.1.2.2, which includes the coordinates of the shoulder, elbow, and wrist along with the classified exercise stage, is stored in a list. This data is then saved to a CSV file named `bicep_curl_data.csv` for further analysis. During the collection process, the pose landmarks are drawn on the video feed to provide visual feedback, and the data collection continues until the user decides to stop by pressing the 'q' key.

```
In [2]: 1 import tensorflow as tf
2 import pandas as pd
3 from sklearn.model_selection import train_test_split
4 from sklearn.preprocessing import LabelEncoder
5 from sklearn.preprocessing import StandardScaler
6
7 # Load data
8 data = pd.read_csv('bicep_curl_data.csv')
9
10 # Prepare data
11 X = data[['shoulder_x', 'shoulder_y', 'elbow_x', 'elbow_y', 'wrist_x', 'wrist_y']].values
12 y = data['stage'].values
13
14 # Encode Labels
15 label_encoder = LabelEncoder()
16 y_encoded = label_encoder.fit_transform(y)
17
18 # Normalize features
19 scaler = StandardScaler()
20 X_scaled = scaler.fit_transform(X)
21
22 # Split data
23 X_train, X_test, y_train, y_test = train_test_split(X_scaled, y_encoded, test_size=0.2, random_state=42)
24
25 # Build model
26 model = tf.keras.Sequential([
27     tf.keras.layers.Dense(64, activation='relu', input_shape=(X_train.shape[1],)),
28     tf.keras.layers.Dense(32, activation='relu'),
29     tf.keras.layers.Dense(2, activation='softmax') # Binary classification: up or down
30 ])
31
32 model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
33
34 # Train model
35 model.fit(X_train, y_train, epochs=10, validation_split=0.1)
36
37 # Evaluate model
38 loss, accuracy = model.evaluate(X_test, y_test)
39 print(f'Accuracy: {accuracy}')
40
41 # Save model
42 model.save('bicep_curl_model.h5')
43
```

Epoch 1/10

Figure 5.1.6.1 (a): Conversion of Trained Model to TensorFlow Lite Format


```

Epoch 1/10
D:\Anaconda\Lib\site-packages\keras\src\layers\core\dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument
to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
super().__init__(activity_regularizer=activity_regularizer, **kwargs)
94/94 ----- 1s 4ms/step - accuracy: 0.7944 - loss: 0.4849 - val_accuracy: 0.9401 - val_loss: 0.1813
Epoch 2/10
94/94 ----- 0s 2ms/step - accuracy: 0.9446 - loss: 0.1662 - val_accuracy: 0.9970 - val_loss: 0.0756
Epoch 3/10
94/94 ----- 0s 2ms/step - accuracy: 0.9958 - loss: 0.0659 - val_accuracy: 0.9970 - val_loss: 0.0275
Epoch 4/10
94/94 ----- 0s 2ms/step - accuracy: 0.9990 - loss: 0.0269 - val_accuracy: 0.9970 - val_loss: 0.0128
Epoch 5/10
94/94 ----- 0s 2ms/step - accuracy: 0.9968 - loss: 0.0166 - val_accuracy: 0.9970 - val_loss: 0.0080
Epoch 6/10
94/94 ----- 0s 2ms/step - accuracy: 0.9992 - loss: 0.0099 - val_accuracy: 1.0000 - val_loss: 0.0053
Epoch 7/10
94/94 ----- 0s 2ms/step - accuracy: 0.9981 - loss: 0.0082 - val_accuracy: 1.0000 - val_loss: 0.0047
Epoch 8/10
94/94 ----- 0s 2ms/step - accuracy: 0.9988 - loss: 0.0084 - val_accuracy: 1.0000 - val_loss: 0.0030
Epoch 9/10
94/94 ----- 0s 2ms/step - accuracy: 0.9990 - loss: 0.0063 - val_accuracy: 1.0000 - val_loss: 0.0025
Epoch 10/10
94/94 ----- 0s 2ms/step - accuracy: 0.9995 - loss: 0.0041 - val_accuracy: 1.0000 - val_loss: 0.0022
27/27 ----- 0s 2ms/step - accuracy: 0.9979 - loss: 0.0071

```

Figure 5.1.6.1 (b): Conversion of Trained Model to TensorFlow Lite Format

First, as shown in the figure above, the collected dataset, `bicep_curl_data.csv`, is loaded into the program. This dataset contains positional coordinates for the shoulder, elbow, and wrist, as well as the stage of the exercise (either "up" or "down").

The next step involves preparing the data for training. The positional coordinates are extracted as features (denoted as `X`), and the exercise stages are labeled as `y`. To ensure the model performs well, the exercise stages are encoded into numerical labels using `LabelEncoder`. The feature values are normalized using `StandardScaler` to bring them into a standard range, which helps in improving the model's performance.

The dataset is then split into training and test sets using `train_test_split`. This separation allows the model to be trained on one portion of the data while evaluating its performance on another portion that it hasn't seen during training.

Then a neural network model is built using TensorFlow's Keras API. The model consists of three layers:

1. A dense layer with 64 neurons and ReLU activation function.
2. A second dense layer with 32 neurons and ReLU activation.
3. An output layer with 2 neurons and a softmax activation function, which is suitable for binary classification (distinguishing between "up" and "down").

The model is compiled using the Adam optimizer and sparse categorical cross-entropy loss function, which is appropriate for this classification task. The model is trained for 10 epochs, with a portion of the training data reserved for validation to monitor its performance during training.

After training, the model's performance is evaluated on the test set to assess its accuracy. The final step is to save the trained model to a file named `bicep_curl_model.h5`, which can be used later for making predictions or further analysis.

```
In [3]: 1 import tensorflow as tf
        2
        3 # Load the trained model
        4 model = tf.keras.models.load_model('bicep_curl_model.h5')
        5
        6 # Convert the model to TensorFlow Lite format
        7 converter = tf.lite.TFLiteConverter.from_keras_model(model)
        8 tflite_model = converter.convert()
        9
        10 # Save the TFLite model
        11 with open('bicep_curl_model.tflite', 'wb') as f:
        12     f.write(tflite_model)
        13
```

Figure 5.1.6.2: Save model

The model is converted to a .tflite (TensorFlow Lite) format. This process ensures that the model can run efficiently on Android devices.

5.2 System Operation

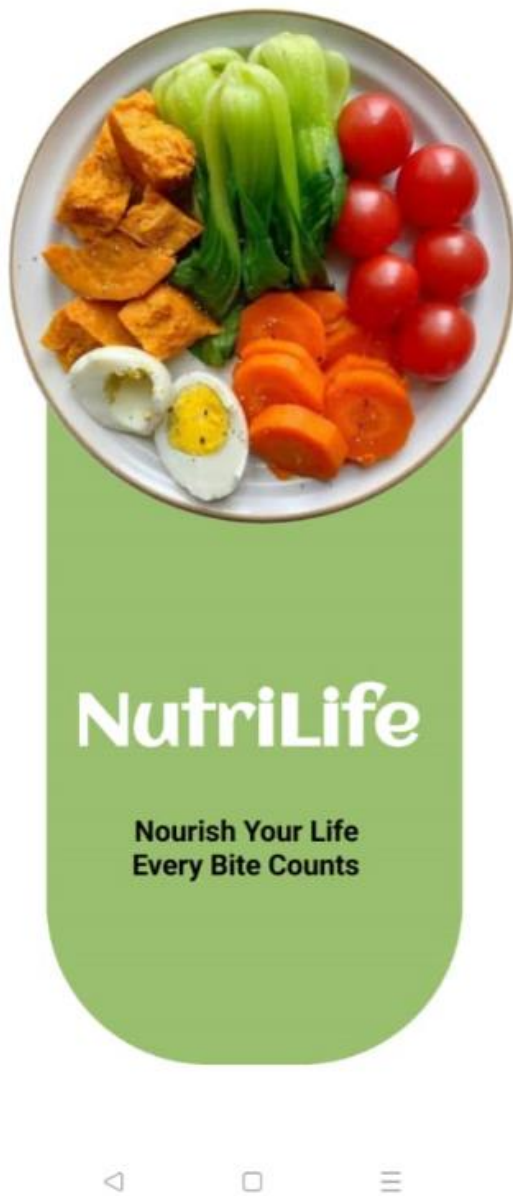


Figure 5.2.1: Splash Screen

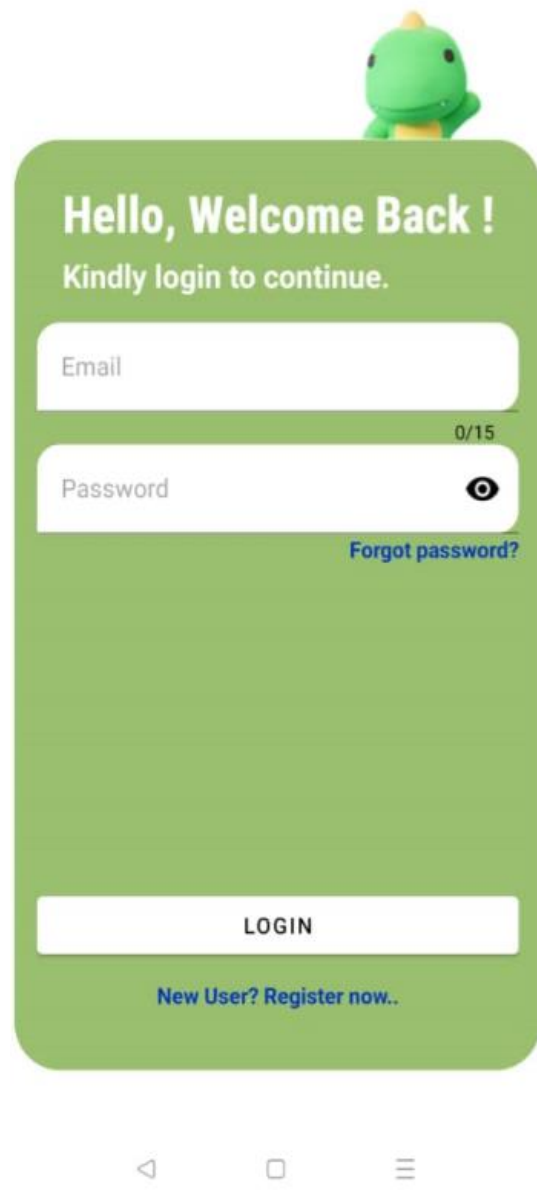


Figure 5.2.2: Login Page

Figure 5.2.1 shows the splash screen of NutriLife, when users launch the application, this splash screen will display the application name which is “NutriLife,” and also the application slogan which is “Nourish your life, every bite counts,” which is to indicate the application is loading.

After the application finished loading, it will navigate to the first page which is the Login page as shown in Figure 5.2.2. Users are required to input their Username and Password which will then navigate to the home page of the application.

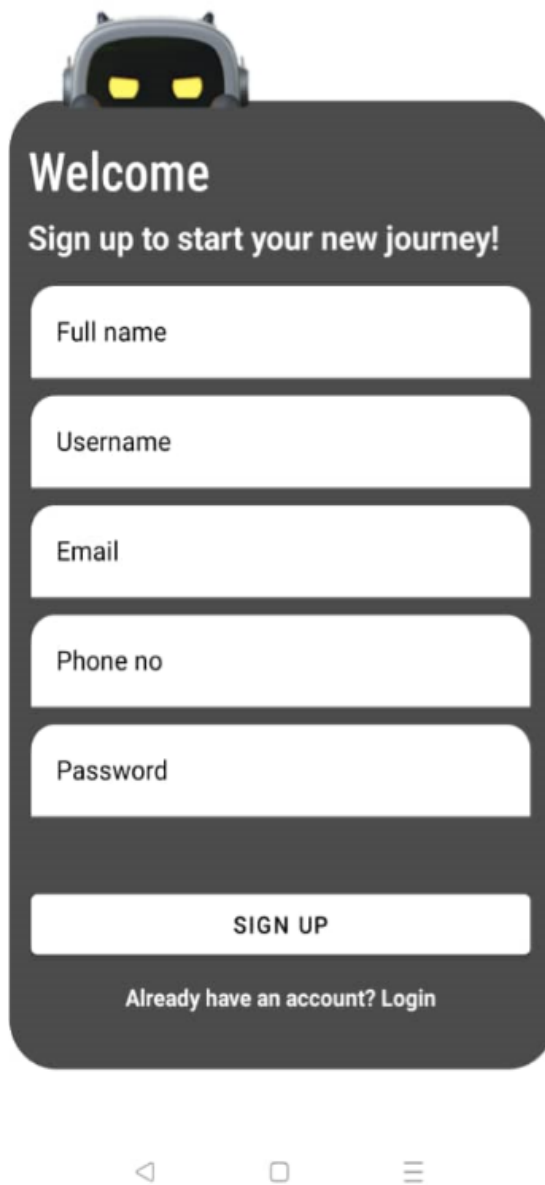


Figure 5.2.3: Sign up Page

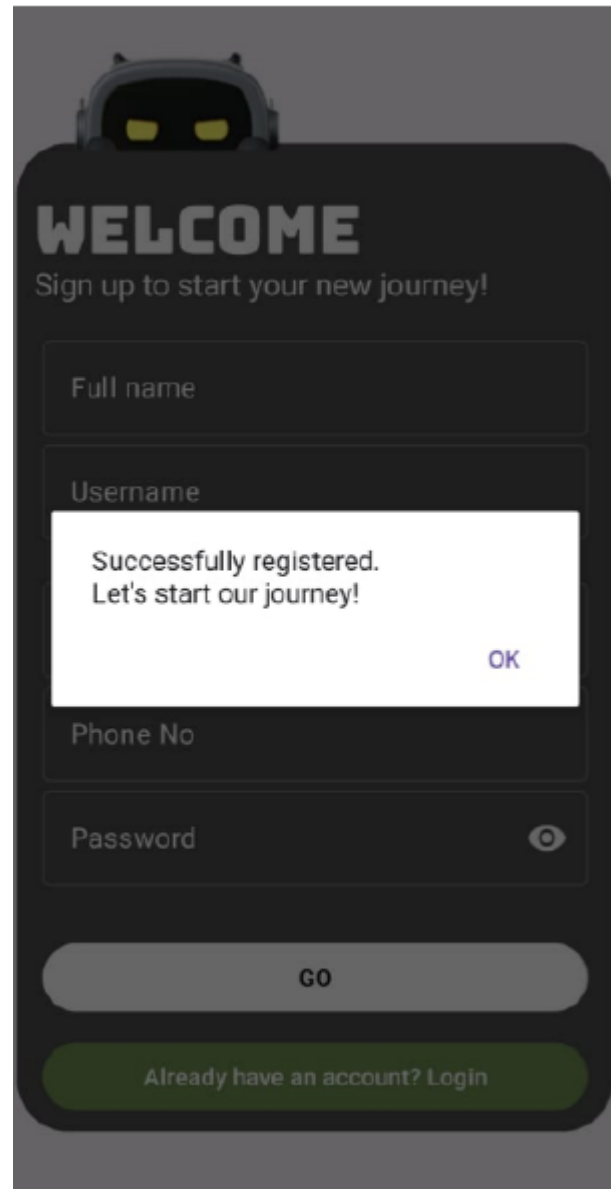


Figure 5.2.4: Finished registered

If the user is a first-time user, they required to sign up by filling up the necessary information such as full name, username which max length is 15, email address, phone number and also set their password. The system will then verify the email, password to ensure both are unique and have not been stored in the database.

After the user has successfully registered, it will pop out a message box to let the users to know that they have registered successfully.



Figure 5.2.5: Introduction of the main highlighted features in NutriLife



Figure 5.2.6: Introduction of the main highlighted features in NutriLife



Figure 5.2.7: Introduction of the main highlighted features in NutriLife



Figure 5.2.8: Introduction of the main highlighted features in NutriLife

After the user has finished registered, the application will navigate to the Introduction page to let the new user get know of the main highlighted features that they can be used in NutriLife. User can click on the “Start” button to navigate to homepage.

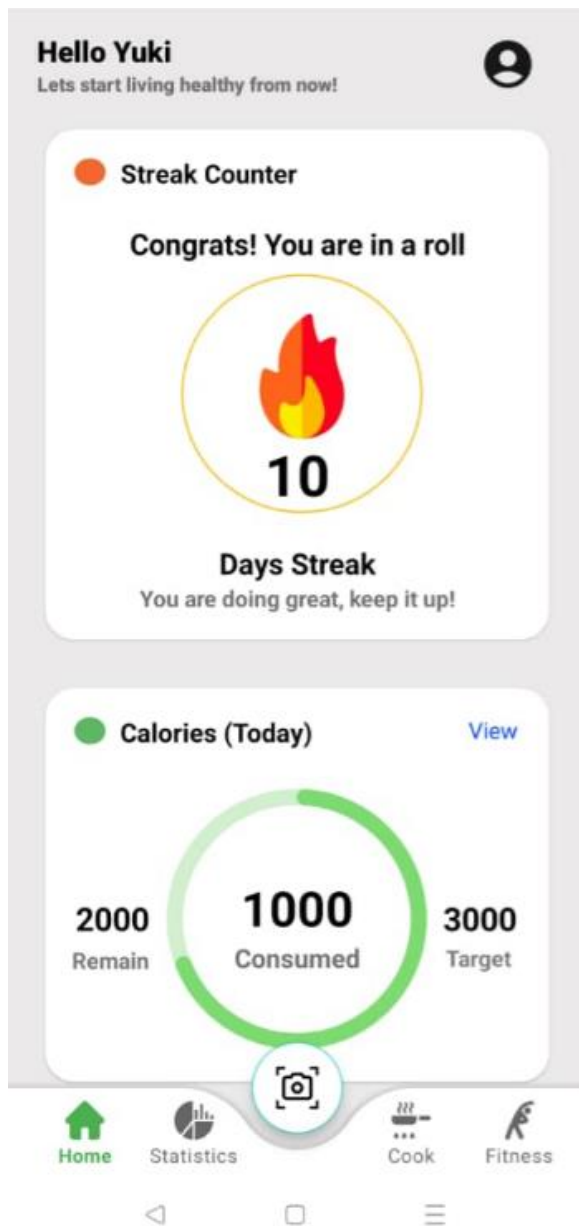


Figure 5.2.9: Homepage

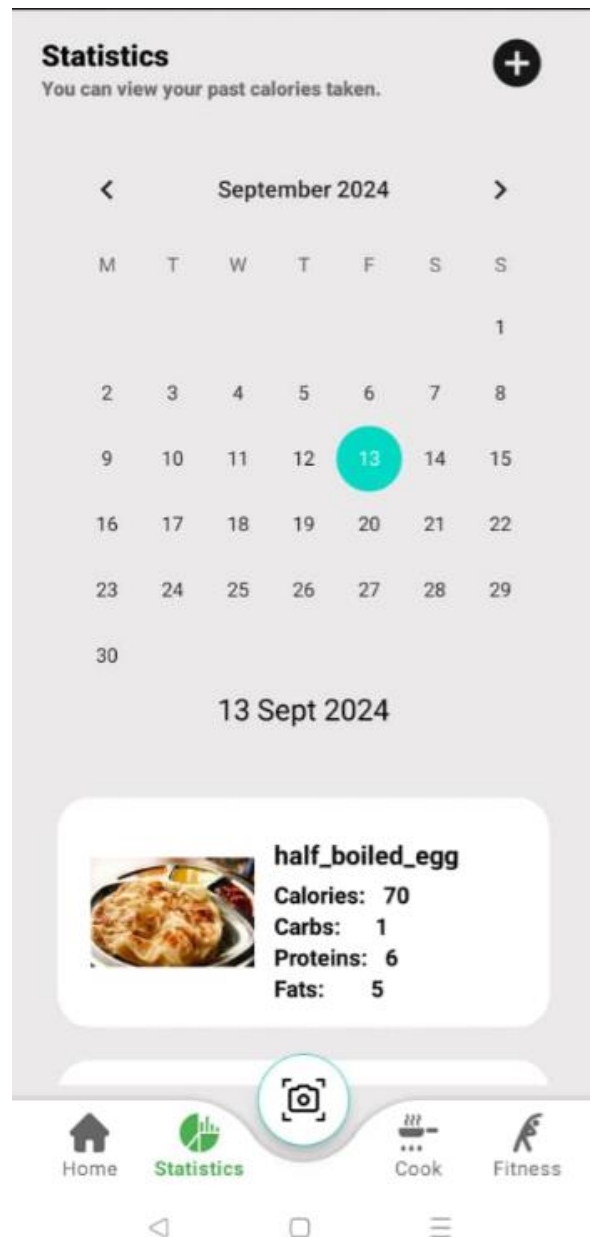


Figure 5.2.10: Statistic Page

The home page of the application provides users with a comprehensive overview of their fitness activities. At a glance, users can see their **Streak Counter**, which tracks the number of consecutive days they have met their fitness goals. This feature serves as a motivational tool to help maintain consistency in their routines. Additionally, users are presented with details on

Today's Calories, showing the total calories consumed for the current day. Alongside this, the **Remaining Calories** feature displays how many calories are left for the day based on their set calorie target, enabling users to manage their intake effectively. Finally, the **Calorie Target** provides a clear goal for daily calorie consumption, assisting users in staying on track with their dietary objectives.

The static page of the application features an interactive calendar that allows users to select different dates. Upon selecting a date, users can view food records from the Food Info page associated with that specific day. This functionality provides users with the ability to review historical dietary data, making it easier to track their food intake over time. By accessing past food records, users can analyze their eating patterns and make informed adjustments to their diet, enhancing their overall fitness and health management.



Figure 5.2.11: Cook Page

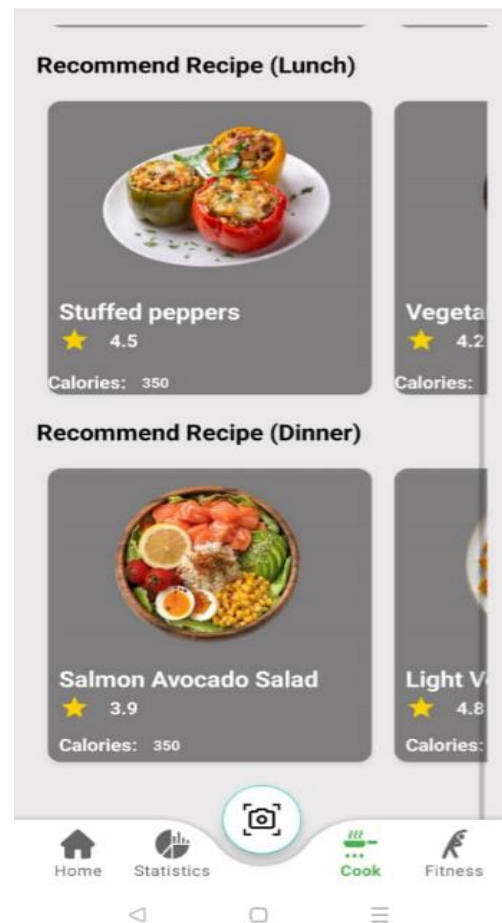


Figure 5.2.12: Cook Page



Figure 5.2.13: Example of detailed information of recommended meal for breakfast

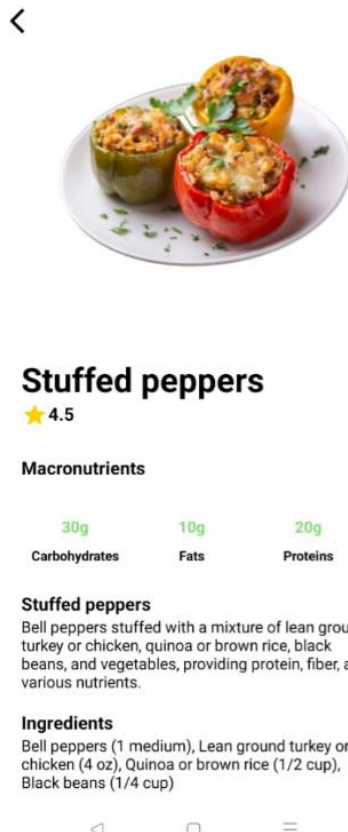


Figure 5.2.14: Example of detailed information of recommended meal for lunch



Figure 5.2.15: Example of detailed information of recommended meal for dinner

The Cook page as shown in figure 5.2.11 and figure 5.2.12 offers users a selection of simple and easy-to-cook recommended meals. This page provides detailed information on each recipe to help users prepare nutritious and delicious dishes. For each recommended meal as shown from figure 5.2.13 to figure 5.2.15, users can view the **Macronutrient Breakdown**, which includes portions of proteins, fats, and carbohydrates to help them balance their diet. The **Meal Description** offers a brief overview of the dish, highlighting its key features and benefits. Additionally, the page includes a list of **Ingredients** required for the recipe, along with step-by-step **Cooking Instructions**. This structured presentation ensures that users can follow the recipe with ease, making meal preparation straightforward and enjoyable while supporting their health and fitness goals.

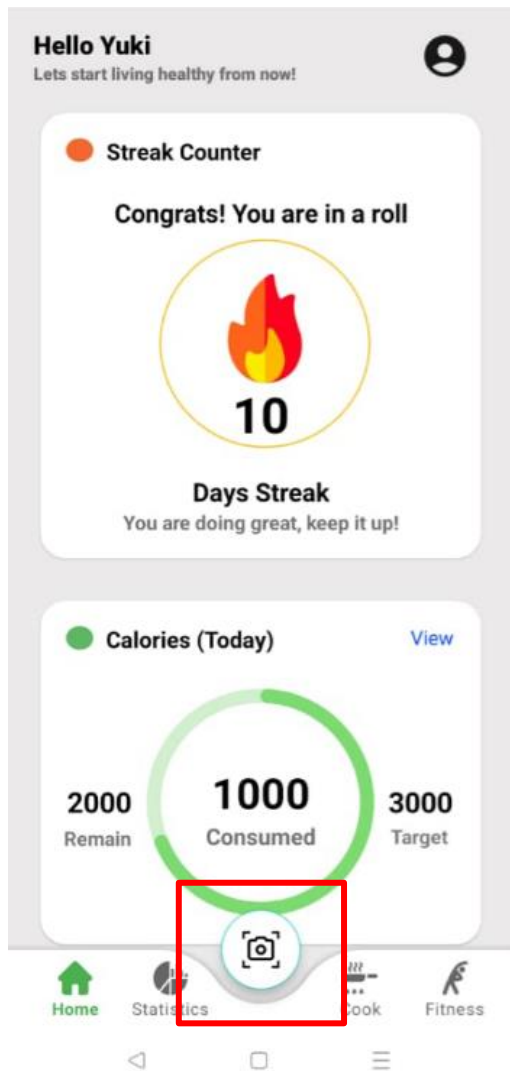


Figure 5.2.16 (a): Food recognition page

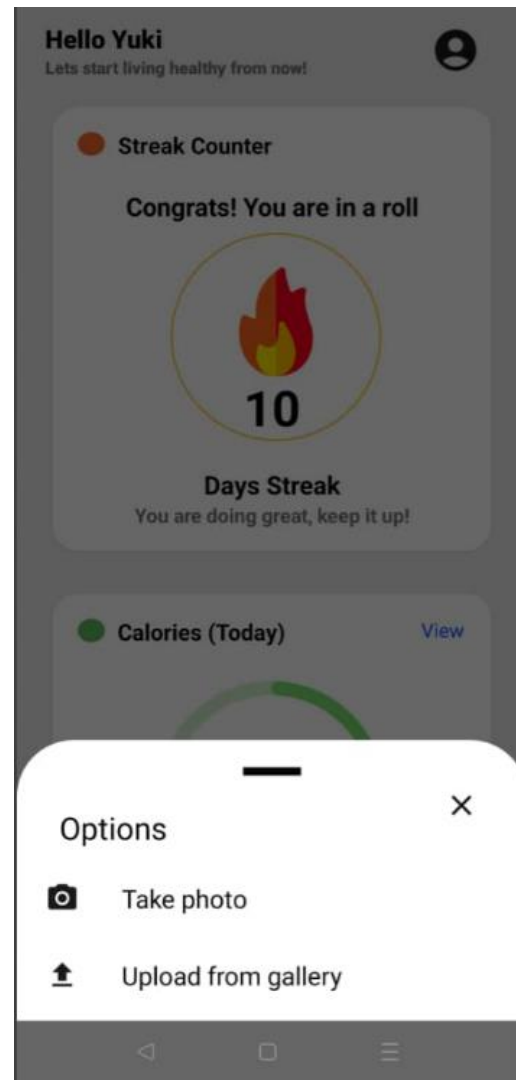


Figure 5.2.16 (b): Food recognition page

On the Food Recognition page, users can initiate the food identification process by interacting with a camera icon. Clicking this icon triggers a pop-up dialog that offers two options: **Take a Photo** or **Upload from Gallery** as shown in the figure 5.2.16 (b).

- **Take a Photo:** Users can use their device's camera to capture an image of the food they wish to identify. This option allows for immediate food recognition based on the freshly taken photo as shown in the figure 5.2.17.
- **Upload from Gallery:** Users can select an image from their device's gallery. This option is ideal for identifying food items from previously captured photos as shown in figure 5.2.18 (a) and figure 5.2.18 (b).

CHAPTER 5

Once the user chooses an option, the selected image is processed to recognize the food, and the results are displayed on the Food Info page, providing users with useful information about the identified food item as shown in figure 5.2.19.

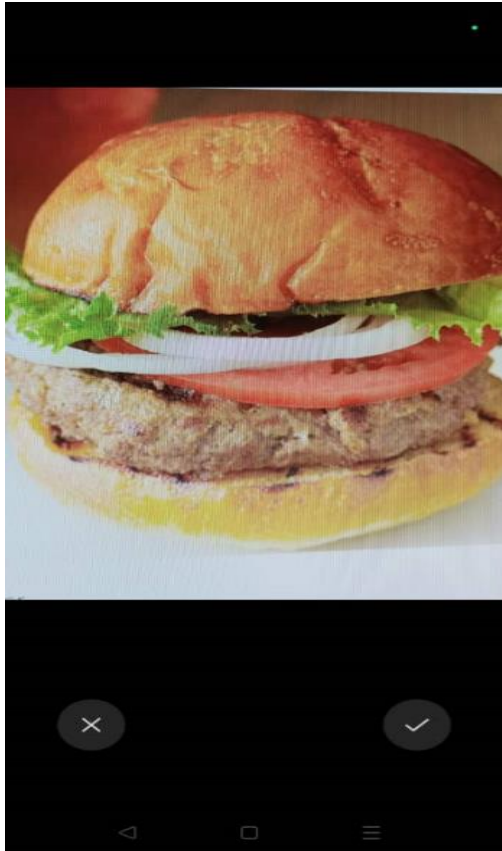


Figure 5.2.17: Take photo options

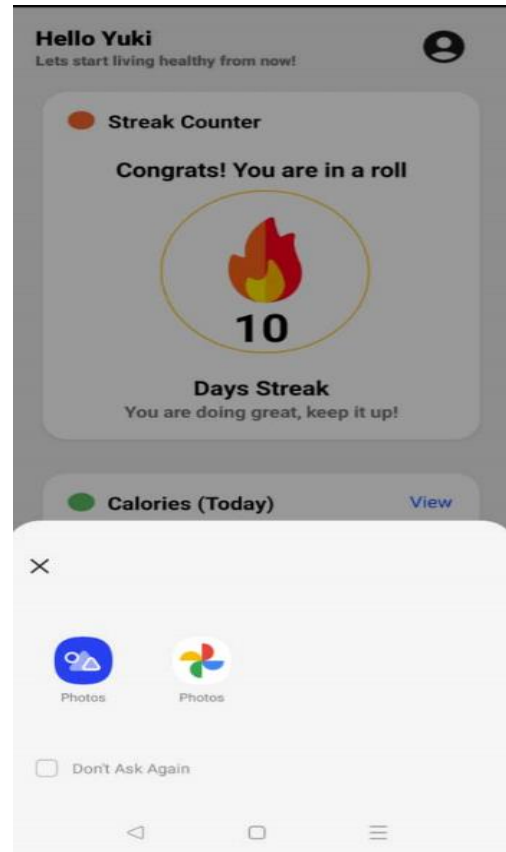


Figure 5.2.18 (a): Upload from gallery options

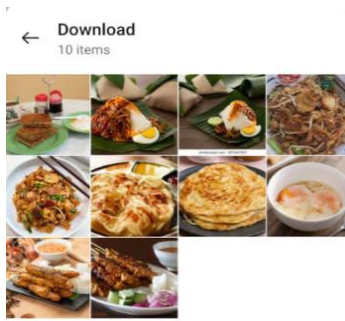


Figure 5.2.18 (b): Upload from gallery options



Figure 5.2.19: Calories and macronutrients of the food uploaded

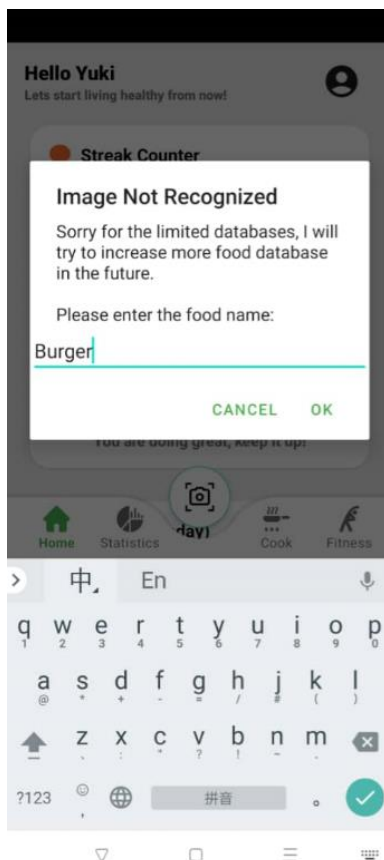


Figure 5.2.20: Food not recognized

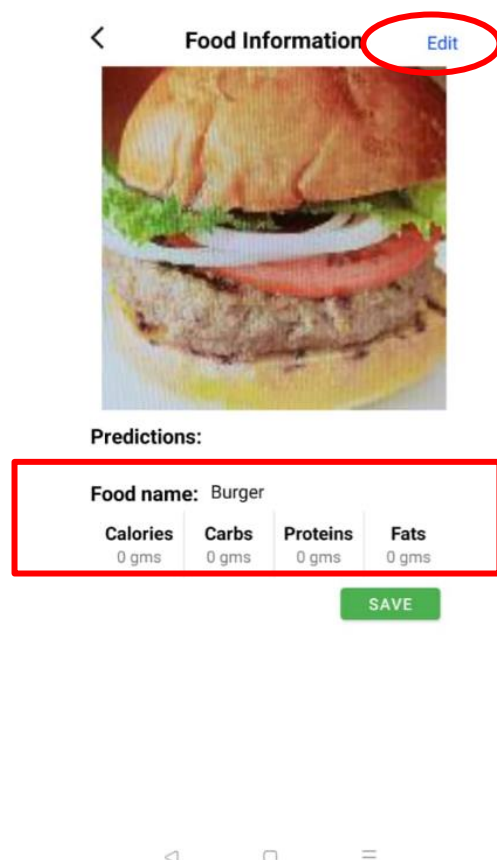


Figure 5.2.21: Food info page

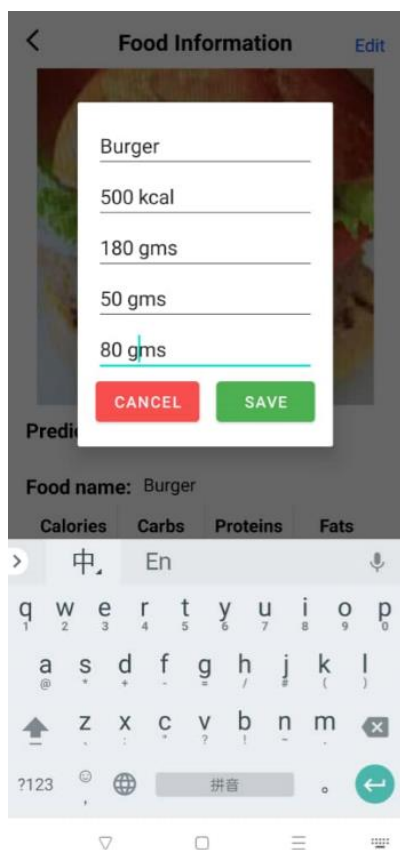


Figure 5.2.22: Edit the food's information

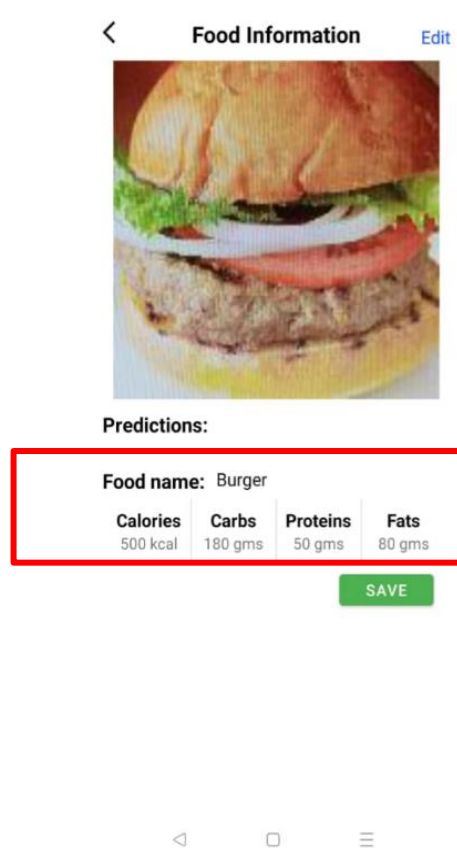


Figure 5.2.23: The information of the food is being updated

If the food recognition process fails to identify the food from the taken or uploaded photo, a pop-up dialog will appear prompting the user to manually input the food name. This feature ensures that the user can still log and track their food intake even when the automated recognition does not work as shown in Figure 5.2.20.

After entering the food name as shown in Figure 5.2.20, the user is redirected to the Food Info page as shown in figure 5.2.21. On this page, the food item will be listed with default values for calories and macronutrients set to zero by default. Users can update these details by clicking the **Edit** button as shown in figure 5.2.21. As shown in figure 5.2.22, this edits functionality allows users to:

- **Add or Adjust Calories:** Enter the correct calorie count for the food item.
- **Specify Macronutrients:** Input the amounts of proteins, fats, and carbohydrates.
- **Modify Food Name:** Change the food name if necessary.

The user will get the updated information as shown in figure 5.2.23 after the user save their edited information

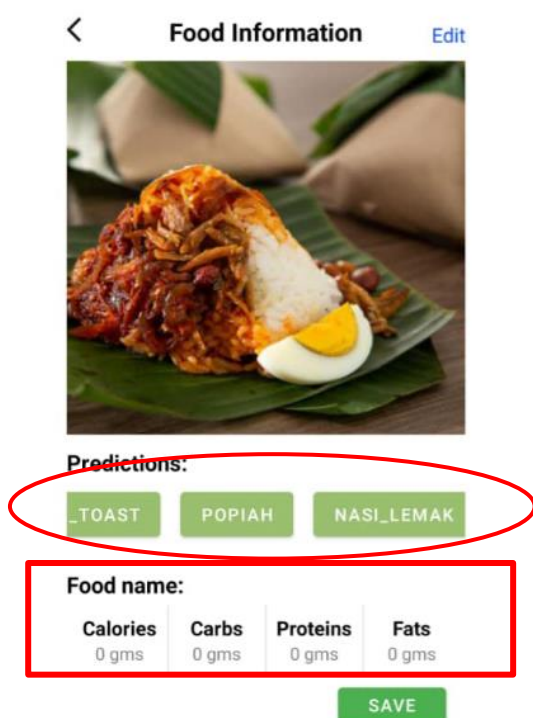


Figure 5.2.24: Predicted food based on the food taken or uploaded, values of the calories and macronutrients are set to zero by default

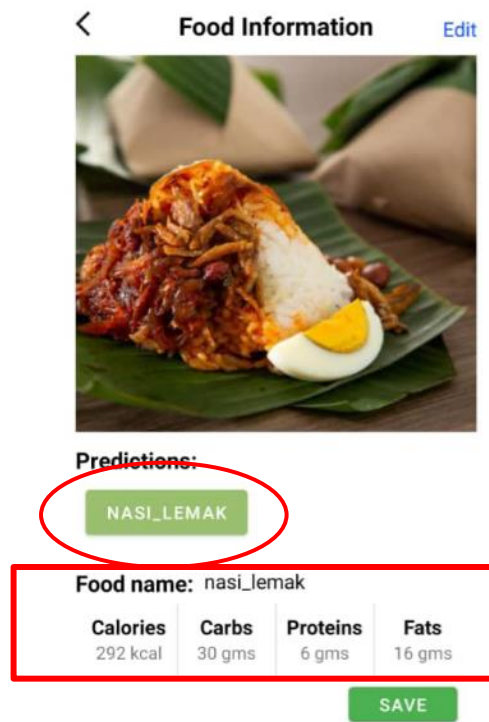


Figure 5.2.25: Predicted food based on the food taken or uploaded, values of the calories and macronutrients are set to the average estimated values

When the food recognition feature successfully identifies a food item from the photo or upload, a pop-up window will display the top three predicted food items. This allows users to select the correct food from the list of predictions.

Initially, the food’s nutritional values such as calories and macronutrients are set to zero by default. Upon the user selecting one of the predicted food items, these values are updated automatically. The system calculates and displays the estimated average nutritional values based on the selected food item. This feature ensures that the food’s nutritional information is accurately reflected in the user’s records without manual entry, enhancing the efficiency of food tracking.

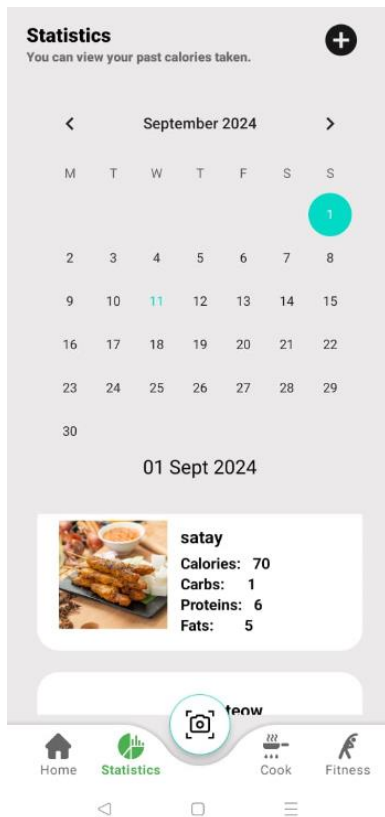


Figure 5.27 (a): Example of food records saved on 1st September 2024

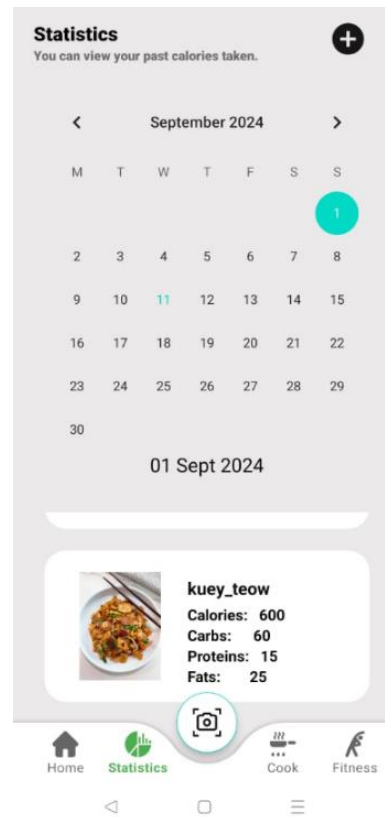


Figure 5.27 (b): Example of food records saved on 1st September 2024

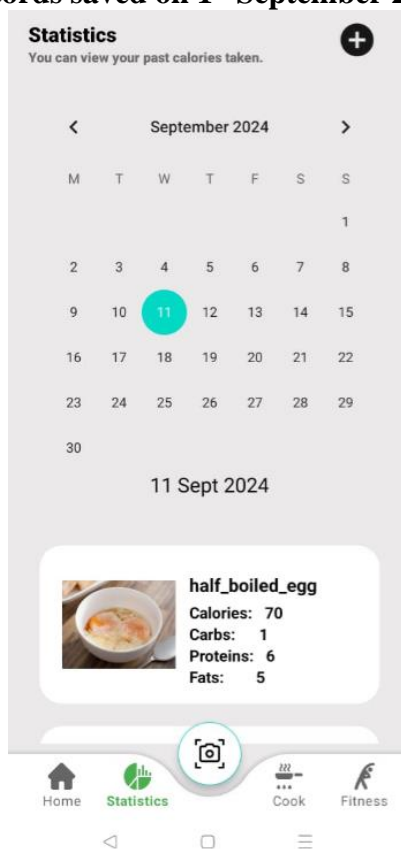


Figure 5.28 (a): Example of food records saved on 11th September 2024

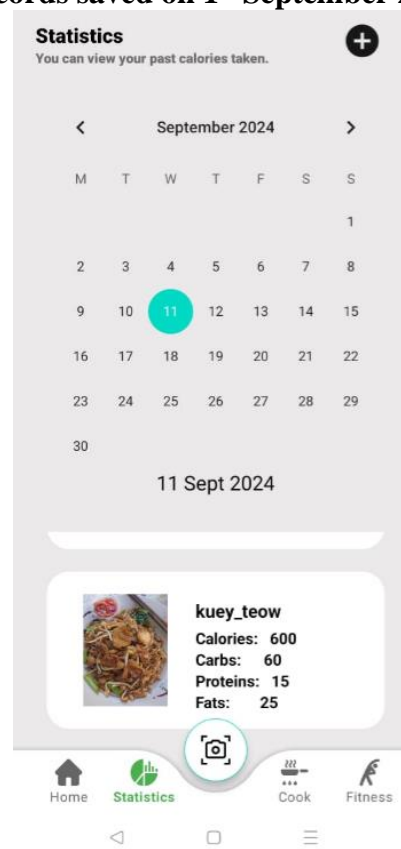


Figure 5.28 (b): Example of food records saved on 11th September 2024

After selecting the appropriate food from the top predictions and clicking the save button, if the user chooses not to modify any information, the food record is immediately saved to the database. This action ensures that the food entry is recorded with its corresponding nutritional details.

Users can later review their saved food records on the Statistics page. Here, food records are displayed according to the selected date, reflecting the entries recorded by the user on that specific day. This feature allows for convenient tracking and analysis of dietary habits, providing a clear view of nutritional intake over time as shown in figure 5.2.27 and figure 5.2.28 .

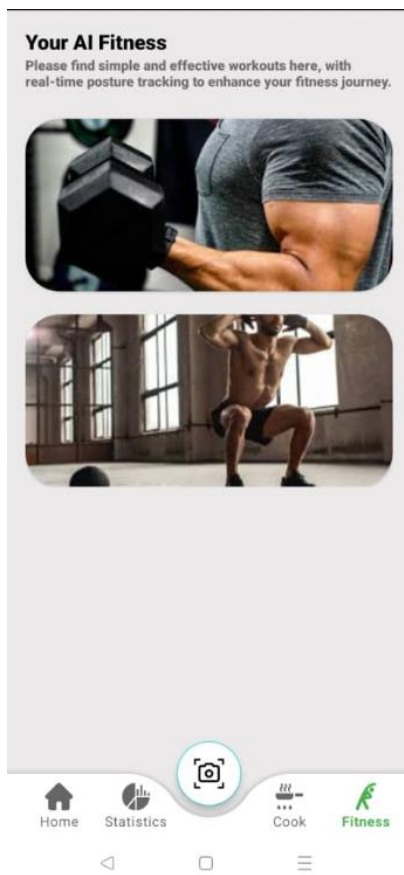


Figure 5.2.29: Fitness Page



Figure 5.2.30: Squat



Figure 5.2.31: Bicep curl

On the Fitness Tracking page, users are presented with the option to choose between two exercise tracking modes: **Bicep Curl** or **Squat**. This selection allows users to monitor and record their performance for each specific exercise.

When users select **Bicep Curl**, they can track their progress by capturing data related to their bicep curl form and repetitions. Similarly, selecting **Squat** enables users to track their squat performance, including details such as the angle of the squat and the number of repetitions. Each exercise mode provides tailored feedback and performance metrics to help users achieve their fitness goals effectively as shown in figure 5.2.32 and figure 5.2.33.

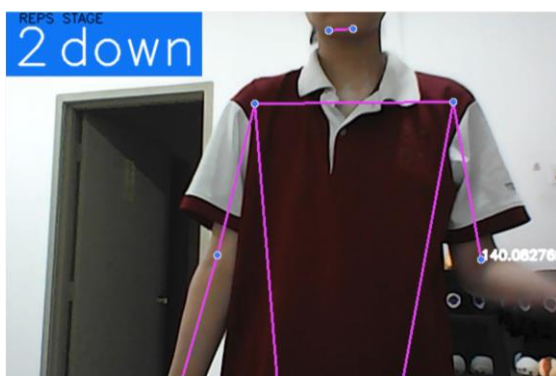


Figure 5.2.32 (a) – Bicep curl

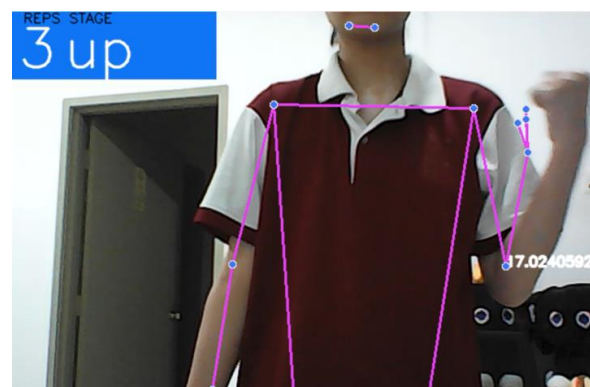


Figure 5.2.32 (a) – Bicep curl



Figure 5.2.33 (a) – Squat

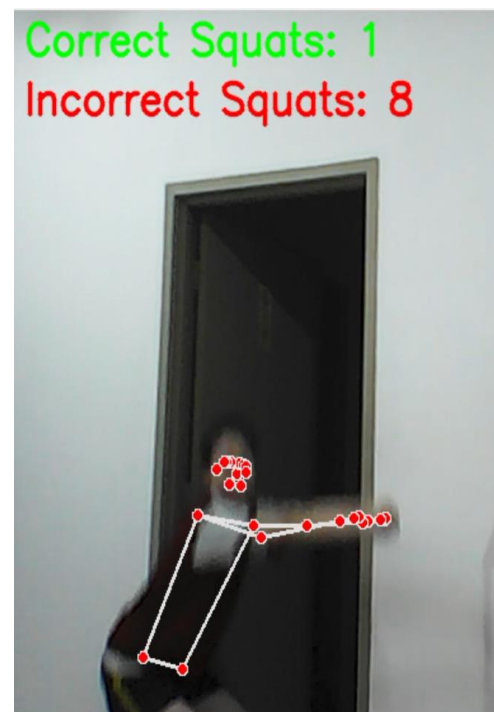


Figure 5.2.33 (b) – Squat

<

Gender

Female Male

Age

24

Height (cm)

160

Weight (kg)

55

Calculate

Figure 5.2.34: BMI page

<

Gender

Female Male

Age

20

Height (cm)

160

Weight (kg)

43

Calculate

Your result

16.8 kg/m² - Underweight

Figure 5.2.35: BMI - underweight

<

Gender

Female Male

Age

20

Height (cm)

160

Weight (kg)

50

Calculate

Your result

19.5 kg/m² - Normal

Figure 5.2.36: BMI page – normal weight

<

Gender

Female Male

Age

20

Height (cm)

160

Weight (kg)

70

Calculate

Your result

27.3 kg/m² - Overweight

Figure 5.2.37: BMI page - overweight

CHAPTER 5

On the BMI (Body Mass Index) page, users can input key personal details such as their gender, age, height in centimetres, and weight in kilograms. Based on these inputs, the app calculates the user's BMI and determines whether they fall into one of three categories: underweight, normal weight, or overweight. This feature provides users with personalized insights into their body weight status and overall health, helping them understand and manage their fitness and nutritional goals.

CHAPTER 6 SYSTEM EVALUATION AND DISCUSSION

6.1 Model Training and Performance

6.1.1 Food Recognition

```

[37] loss, accuracy = model.evaluate(test_generator)
print(f"Test accuracy: {accuracy:.2f}")

/usr/local/lib/python3.10/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121: UserWarning: Your 'PyDataset' class should call 'super().__init__(**kwargs)' in its constructor
self.warn_if_super_not_called()
**** 99a 70dens/step - accuracy: 0.7517 - loss: 0.7454
Test accuracy: 0.74
  
```

Figure 6.1.1.0: The test accuracy of food recognition model (0.74)

As shown in the figure above, the model achieved an accuracy of 74% on the test dataset. This performance metric indicates how well the model generalizes to new, unseen data after training.

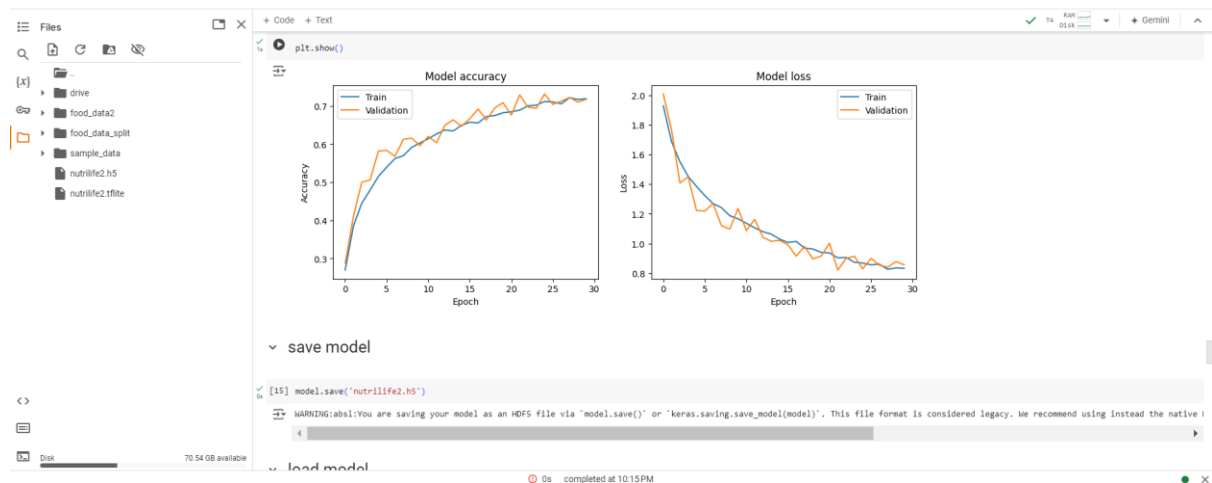


Figure 6.1.1.1: Graph of model accuracy and model loss

The figure 5.1.4.15 illustrates the impact of various factors such as the number of datasets, dense layers, and batch size on the model's accuracy. To ensure robust results and mitigate issues such as overfitting or underfitting, each factor was tested at least 10 times as shown in section 3.1.2.1. In the graphical representation:

- The **blue line** represents the accuracy of the training datasets.
- The **orange line** represents the accuracy of the validation datasets.
- **Model Accuracy:** This plot shows the accuracy of both the training and validation datasets across epochs. It helps in assessing how well the model is learning and generalizing.
- **Model Loss:** This plot illustrates the loss values for both the training and validation datasets. It provides insights into how the model's error decreases over time, reflecting its learning process.

A smaller gap between these two lines indicates that the model generalizes well to new, unseen data, rather than performing well only on the training data.

The blue lines in the graph show the performance metrics for the training dataset throughout the training process. These lines demonstrate how the model's accuracy evolves as it learns from the training data over multiple epochs. A steady and upward slope in the blue lines indicates that the model is successfully learning and improving its accuracy on the training set.

In addition to evaluate the model's classification performance in detail, a confusion matrix was generated. This analysis provides insights into the model's accuracy across different classes and highlights any areas of confusion.

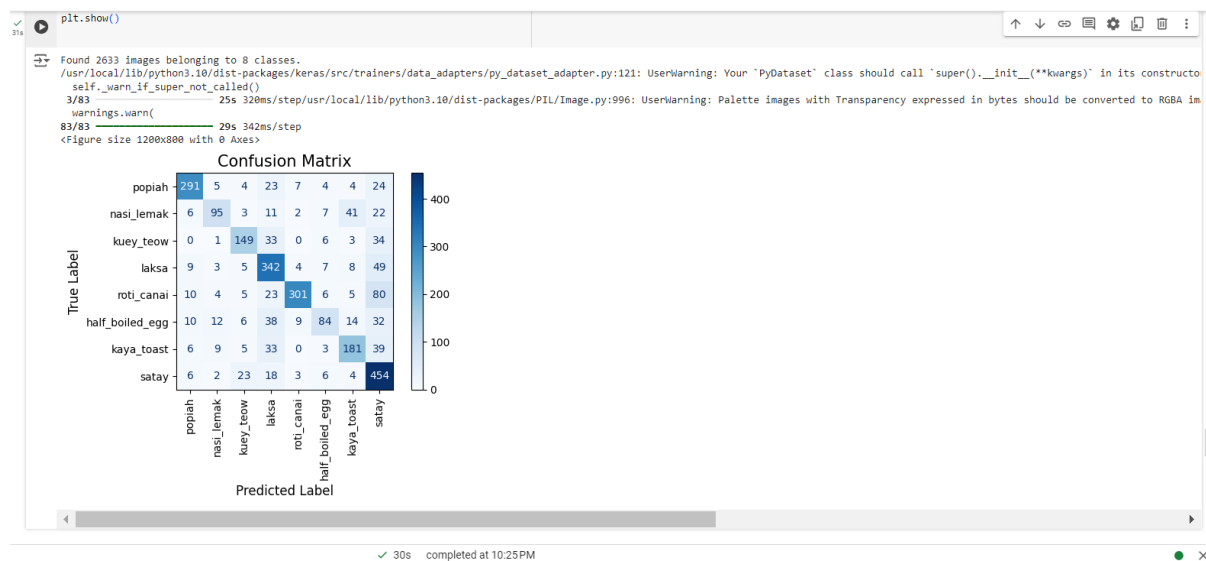


Figure 6.1.1.2: The confusion matrix

As illustrated in the Figure 6.1.1.2, the confusion matrix has shown that:

- **Popiah:** Most 'popiah' instances are correctly identified (291 correct predictions), but there are notable confusions with 'kaya_toast' (24 instances) and 'laksa' (23 instances).
- **Nasi Lemak:** Often correctly predicted (95 instances), but frequently confused with 'popiah' and 'kuey_teow'.
- **Kuey Teow:** Shows some confusion, particularly with 'laksa' (33 instances).
- **Laksa:** Well-predicted (342 instances), although with some confusion.
- **Roti Canai:** Good prediction (301 instances), with notable confusion with 'popiah' and 'laksa'.
- **Half Boiled Egg:** This class seems to be one of the more challenging ones for the model, with substantial misclassifications across multiple other dishes.

- **Kaya Toast:** Mostly well-predicted (181 instances), yet frequently confused with 'popiah'.
- **Satay:** Predominantly correctly classified (454 instances), showing good model performance for this class.

In short, the model generally performs well but struggles with some specific classes, particularly those with visual similarities or less distinct features. This information can be used to further refine the model, improve data quality, or enhance feature extraction to address these misclassification issues.

6.1.2 Fitness tracking

6.1.2.1 Squat



```

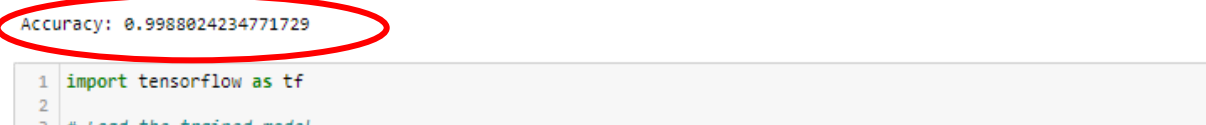
Test accuracy: 0.9969450235366821

In [3]: 1 import tensorflow as tf
        2
  
```

Figure 6.1.2.1.0: Test accuracy of Squat model (0.9969)

As shown in the figure 6.1.2.1.0, the test accuracy for the squat model achieved a 99% accuracy.

6.1.2.2 Bicep curl



```

Accuracy: 0.99888024234771729

1 import tensorflow as tf
2
3 # Load the trained model
  
```

Figure 6.1.2.2.0: Test accuracy of Bicep Curl model (0.9988)

As shown in the figure 6.1.2.2.0, the test accuracy for the squat model achieved a 99% accuracy.

6.2 System performance

The purpose of this verification plan is to verify whether the functionalities implemented in the proposed project have meet the desired result.

Splash screen

Test Action	Expected Result	Result (x/✓)
The user launches the application.	They application will display the animated splash screen when the application is launching	✓

Login Screen

Test Action	Expected Result	Result (x/✓)
The user can input the username and password.	The application will navigate to the home screen.	✓
The user can click on the “New User? Sign Up” button.	The application will navigate to the sign-up screen.	✓

Sign up Screen

Test Action	Expected Result	Result (x/✓)
The user can input the full name, username, email, phone no and password.	The application will pop a message “Successfully registered. Let’s start our journey!”, indicating that the user has registered an account successfully.	✓
The user can click on the “Already have an account? Login” button if user is an existing user”	The application will navigate to the login screen.	✓

Intro Page

Test Action	Expected Result	Result (*/✓)
After user has successfully registered an account.	The application will display the main features of the application.	✓
The user clicks on the Start button.	The application will navigate to the home page.	✓

Home Page

Test Action	Expected Result	Result (*/✓)
The user scrolls through the home page.	The application will let the user to view the information such as Today's calories consumption and streak count.	✓
The user clicks on the Food button.	The application will navigate to the food recommendation page for breakfast, lunch, and dinner.	✓
The user clicks on the BMI button.	The application will navigate to the BMI page.	✓
The user clicks on the Camera button.	The application will navigate to the Food Recognition page.	✓
The user clicks on the Statistic button.	The application will navigate to the Statistic page.	✓
The user clicks on the Fitness button.	The application will navigate to the Fitness page.	✓

Body Mass Index (BMI) Page

Test Action	Expected Result	Result (*/✓)
The user can input the weight (kilograms) and height (centimeters).	The application will display the weight and height input by the user.	✓

CHAPTER 6

The user can click on the “Calculate button.”	The application will display the result of the BMI calculation of the user.	✓
---	---	---

Food Recognition Page

Test Action	Expected Result	Result (x/✓)
The user can click on the “Take Picture” button.	The application will navigate to the camera to allow the user to take picture of the food.	✓
The user can click on the “Upload from Gallery” button.	The application will navigate the user to the gallery to allow user to scan the picture uploaded from the gallery.	✓
After user take picture of a food.	The application will display the recognized and predict the food that recognized in Food Info page.	✓
After user launch a picture from gallery of a food.	The application will display the recognized and predict the food that recognized in Food Info page.	✓
If food is recognized	The application will navigate to Food Info page.	✓
If the food is not recognized	The application will pop a dialog box to allow user to input the food name.	✓

Food Info page

Test Action	Expected Result	Result (x/✓)
If food is recognized	The application will display the top 3 predictions of the food to allow the user to choose and will display the values of calories and macronutrients once user clicks on the predicted food.	

CHAPTER 6

If food is recognized	The application will display the food name same as the food name input by the user in the dialog box.	
The user clicks on the edit button	The application will pop a dialog box to allow user to edit the food name, values of calories, and macronutrients of the food regardless the food is being recognized or not recognized	✓
The user clicks on the save button	The application will save the food information and will display in statistic page.	✓

Statistic Page

Test Action	Expected Result	Result (x/✓)
The user clicks on the selected date	The application will display the statistics such as food image, food name, values of calories and macronutrients of the food saved in Food Info page.	✓

Fitness Page

Test Action	Expected Result	Result (x/✓)
The user clicks on the bicep curl button	The application will allow user to start the device's camera will be activated to track the user's body movements in real-time using Mediapipe Pose Estimation technology.	✓
The user clicks on the squat button	The application will allow user to start the device's camera will be activated to track the user's body movements in real-time using Mediapipe Pose Estimation technology.	✓

CHAPTER 7 CONCLUSION AND RECOMMENDATION

7.1 Conclusion

The NutriLife mobile application has successfully bridged significant gaps in the domain of food recognition and fitness tracking, specifically targeting Malaysian cuisine and localized fitness needs. The project leveraged advanced technologies such as Convolutional Neural Networks (CNN) for accurate food recognition and MediaPipe for real-time posture analysis during workouts. NutriLife empowers users to make informed dietary choices, track their food intake, and improve their fitness routines through personalized recommendations and AI-driven feedback mechanisms.

Through the successful integration of these technologies, NutriLife provides an accessible and user-friendly platform that not only addresses the lack of localized food databases but also enhances the overall user experience in maintaining a balanced diet and fitness regimen. The project's contributions to individual health and wellness, as well as its potential societal impact in promoting healthier lifestyles, are evident in the practical features implemented.

7.2 Recommendations

To further enhance the functionality and user experience of **NutriLife**, several improvements and future directions are recommended:

1. **Expansion of the Food Database:** While the current version of NutriLife focuses primarily on Malaysian cuisine, it is recommended to expand the database to include international dishes. This would cater to a more diverse user base and offer more comprehensive food tracking options.
2. **Integration with Wearable Devices:** Incorporating support for fitness wearables such as smartwatches or fitness bands would allow users to sync real-time data such as heart rate, step count, and calorie expenditure directly with the app, providing a more holistic view of their health and fitness progress.
3. **User Engagement Through Gamification:** To boost user motivation and retention, introducing gamified elements such as achievement badges, daily or weekly challenges, and social sharing features would encourage users to stay consistent with their health and fitness goals.
4. **Enhanced Nutritional Insights:** Introducing a deeper level of nutritional analysis, including tracking micronutrients such as vitamins and minerals, would provide users with more detailed insights into their dietary habits, helping them achieve more balanced nutrition.
5. **AI-Driven Personalization:** Further enhancing NutriLife's personalization features by incorporating AI-driven algorithms that create tailored fitness and meal plans based on the user's progress, health data, and goals could significantly improve the effectiveness of the app. This would ensure that users receive dynamic recommendations as they advance through their fitness and dietary journeys.

By implementing these recommendations, NutriLife can further solidify its position as a comprehensive tool for promoting healthier lifestyles, both locally and globally.

REFERENCES

- [1] M. Definition, “Health Indicators - Malaysia,” pp. 1–12, 2024, [Online]. Available: <https://learnopencv.com/ai-fitness-trainer-using-mediapipe/>
- [2] World Health Organization, *World health sWORLD HEALTH ORGANIZATION - World health statistics 2024. ISBN 9789240094703. tatistics 2024.* 2024.
- [3] K. Dawn, “AI Fitness Trainer - Build Using MediaPipe For Squat Analysis,” 2022, [Online]. Available: <https://learnopencv.com/ai-fitness-trainer-using-MediaPipe/?debug=0>
- [4] Google AI for Developers, “Pose landmark detection guide,” https://ai.google.dev/edge/mediapipe/solutions/vision/pose_landmarker, pp. 10–14, 2024.
- [5] J. Hruthika, P. Krishna Chaitanya, and G. shiva Chaithanya, “Deep Learning Based Human Pose Estimation Using Opencv,” *Novateur Publications International Journal of Innovations in Engineering Research and Technology*, vol. 7, pp. 2394–3696, 2020.
- [6] J. R. L. Lieffers, J. F. Arocha, K. Grindrod, and R. M. Hanning, “Experiences and Perceptions of Adults Accessing Publicly Available Nutrition Behavior-Change Mobile Apps for Weight Management,” *J Acad Nutr Diet*, vol. 118, no. 2, pp. 229-239.e3, 2018, doi: 10.1016/j.jand.2017.04.015.
- [7] E. Agapie, P. A. Areán, G. Hsieh, and S. A. Munson, “A Longitudinal Goal Setting Model for Addressing Complex Personal Problems in Mental Health,” *Proc ACM Hum Comput Interact*, vol. 6, no. CSCW2, 2022, doi: 10.1145/3555160.
- [8] L. Chen, W. Li, X. Cui, Z. Wang, S. Berretti, and S. Wan, “MS-GDA: Improving Heterogeneous Recipe Representation via Multinomial Sampling Graph Data Augmentation,” *ACM Transactions on Multimedia Computing, Communications and Applications*, vol. 20, no. 7, pp. 10–13, 2024, doi: 10.1145/3648620.
- [9] J. R. L. Lieffers, J. F. Arocha, K. Grindrod, and R. M. Hanning, “Experiences and Perceptions of Adults Accessing Publicly Available Nutrition Behavior-Change Mobile Apps for Weight Management,” *J Acad Nutr Diet*, vol. 118, no. 2, pp. 229-239.e3, 2018, doi: 10.1016/j.jand.2017.04.015.
- [10] W. Peng, S. Kanthawala, S. Yuan, and S. A. Hussain, “A qualitative study of user perceptions of mobile health apps,” *BMC Public Health*, vol. 16, no. 1, pp. 1–11, 2016, doi: 10.1186/s12889-016-3808-0.

Appendix

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Trimester 3, Year 3	Study week no.: 2
Student Name & ID: Yuki Tan Lok Yee, 20ACB05030	
Supervisor: Ts Wong Chee Siang	
Project Title: NutriLife: Empowering health with nutrition and diet app	

1. WORK DONE

- Revising FYP1 report
- Reorganize and modify the requirements
- Amend the project background and literature review

2. WORK TO BE DONE

- Need to learn for the CNN layer again
- Need to learn to setup for anaconda environments
- Need to go test in FYP lab

3. PROBLEMS ENCOUNTERED

- The google colab cannot handle large datasets with free access.

4. SELF EVALUATION OF THE PROGRESS

So far so good



Supervisor's signature



Student's signature

POSTER



PLAGIARISM CHECK RESULT

FYP2 for turnitin.docx

ORIGINALITY REPORT

11 %	6 %	7 %	6 %
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Universiti Tunku Abdul Rahman Student Paper	1 %
2	"Innovations and Advances in Cognitive Systems", Springer Science and Business Media LLC, 2024 Publication	<1 %
3	Enachi Andrei, Turcu Cornel, George Culea, Sghera Bogdan Constantin, Ungureanu Andrei Gabriel. "Romanian Sign Language and Mime-Gesture Recognition", International Journal of Advanced Computer Science and Applications, 2024 Publication	<1 %
4	Abdelaziz Testas. "Distributed Machine Learning with PySpark", Springer Science and Business Media LLC, 2023 Publication	<1 %
5	Submitted to University of Westminster Student Paper	<1 %
6	www.mdpi.com Internet Source	<1 %

PLAGIARISM CHECK RESULT

7	"Proceedings of the 12th International Conference on Soft Computing for Problem Solving", Springer Science and Business Media LLC, 2024 Publication	<1%
8	eprints.utar.edu.my Internet Source	<1%
9	eprints.utm.my Internet Source	<1%
10	Submitted to Columbia University Student Paper	<1%
11	iris.who.int Internet Source	<1%
12	Mehdi Ghayoumi. "Generative Adversarial Networks in Practice", CRC Press, 2023 Publication	<1%
13	Submitted to Multimedia University Student Paper	<1%
14	"Proceedings of the International Conference on Internet of Everything and Quantum Information Processing", Springer Science and Business Media LLC, 2024 Publication	<1%
15	Submitted to University of Alabama at Birmingham Student Paper	<1%

PLAGIARISM CHECK RESULT

16	www.ncbi.nlm.nih.gov Internet Source	<1 %
17	Submitted to University of North Texas Student Paper	<1 %
18	Submitted to Student Paper	<1 %
19	Submitted to Federation University Student Paper	<1 %
20	www.codewithc.com Internet Source	<1 %
21	Submitted to Roehampton University Student Paper	<1 %
22	www.ceaa-acee.gc.ca Internet Source	<1 %
23	Chinmay Chakraborty, Manisha Guduri, K. Shyamala, B. Sandhya. "Multifaceted Approaches for Data Acquisition Processing and Communication", CRC Press, 2024 Publication	<1 %
24	Submitted to University of Technology, Sydney Student Paper	<1 %
25	dspace.daffodilvarsity.edu.bd:8080 Internet Source	<1 %
	Submitted to University of Exeter	

PLAGIARISM CHECK RESULT

26	Student Paper	<1 %
27	ijritcc.org Internet Source	<1 %
28	fastercapital.com Internet Source	<1 %
29	Submitted to AUT University Student Paper	<1 %
30	Submitted to Asia Pacific University College of Technology and Innovation (UCTI) Student Paper	<1 %
31	Submitted to Victoria University Student Paper	<1 %
32	Submitted to National School of Business Management NSBM, Sri Lanka Student Paper	<1 %
33	accesstonutrition.org Internet Source	<1 %
34	Haldis Borgen, Oline Zachariassen, Pelin Mise, Ahmet Yildiz, Özlem Özgöbek. "Private Sensitive Content on Social Media: An Analysis and Automated Detection for Norwegian", Linköping University Electronic Press, 2024 Publication	<1 %

PLAGIARISM CHECK RESULT

35	Submitted to Mercer University Student Paper	<1 %
36	Submitted to University of Hertfordshire Student Paper	<1 %
37	Submitted to University of Mauritius Student Paper	<1 %
38	ir.kluniversity.in Internet Source	<1 %
39	Submitted to University of Bedfordshire Student Paper	<1 %
40	Submitted to University of New Mexico System Student Paper	<1 %
41	www.ramp-alberta.org Internet Source	<1 %
42	Submitted to Brunel University Student Paper	<1 %
43	Submitted to Florida International University Student Paper	<1 %
44	Submitted to Middlesex University Student Paper	<1 %
45	Ton Duc Thang University Publication	<1 %
46	Submitted to Southampton Solent University	

PLAGIARISM CHECK RESULT

	Student Paper	<1 %
47	Srishti Verma, Manan Garg, Ronit Gandotra, Prajwal Mahajan et al. "Feed Forward Neural Network-Based Approaches for Vaccine Target Identification: Implementation and Insights", Open Science Framework, 2024 Publication	<1 %
48	s3.eu-central-1.amazonaws.com Internet Source	<1 %
49	Submitted to Heriot-Watt University Student Paper	<1 %
50	Submitted to Westcliff University Student Paper	<1 %
51	"Advanced Computing and Intelligent Technologies", Springer Science and Business Media LLC, 2024 Publication	<1 %
52	Submitted to Harrisburg University of Science and Technology Student Paper	<1 %
53	Submitted to UOW Malaysia KDU University College Sdn. Bhd Student Paper	<1 %
54	mural.maynoothuniversity.ie Internet Source	<1 %

PLAGIARISM CHECK RESULT

55	Submitted to HCUC Student Paper	<1 %
56	Submitted to Jacobs University, Bremen Student Paper	<1 %
57	Jaiteg Singh, S B Goyal, Rajesh Kumar Kaushal, Naveen Kumar, Sukhjit Singh Sehra. "Applied Data Science and Smart Systems - Proceedings of 2nd International Conference on Applied Data Science and Smart Systems 2023 (ADSSS 2023) 15-16 Dec, 2023, Rajpura, India", CRC Press, 2024 Publication	<1 %
58	Submitted to University of Greenwich Student Paper	<1 %
59	Submitted to University of Stirling Student Paper	<1 %
60	ijarsct.co.in Internet Source	<1 %
61	kitakyu.repo.nii.ac.jp Internet Source	<1 %
62	Submitted to Alexandru Ioan Cuza University of Iasi Student Paper	<1 %
63	Submitted to Athlone Institute of Technology Student Paper	<1 %

PLAGIARISM CHECK RESULT

64	Submitted to Letterkenny Institute of Technology Student Paper	<1 %
65	Submitted to Xiamen University Student Paper	<1 %
66	nemo.asee.org Internet Source	<1 %
67	"iCAST 2017 proceedings", 2017 IEEE 8th International Conference on Awareness Science and Technology (iCAST), 2017 Publication	<1 %
68	pdfs.semanticscholar.org Internet Source	<1 %
69	www.ijraset.com Internet Source	<1 %
70	Grace Ataguba, Rita Orji. "Toward the design of persuasive systems for a healthy workplace: a real-time posture detection", Frontiers in Big Data, 2024 Publication	<1 %
71	Submitted to Universiti Teknologi Malaysia Student Paper	<1 %
72	dev.to Internet Source	<1 %
73	ipfs.io Internet Source	

PLAGIARISM CHECK RESULT

		<1 %
74	www.netiq.com Internet Source	<1 %
75	www.openj-gate.com Internet Source	<1 %
76	5dok.net Internet Source	<1 %
77	Arun Kumar Sinha, Abhishek Sharma, Luiz Alberto Pasini Melek, Daniele D. Caviglia. "Smart Embedded Systems - Advances and Applications", CRC Press, 2023 Publication	<1 %
78	Giuseppe Mancia, Guido Grassi, Konstantinos P. Tsioufis, Anna F. Dominiczak, Enrico Agabiti Rosei. "Manual of Hypertension of the European Society of Hypertension", CRC Press, 2019 Publication	<1 %
79	Matt Duckham, Qian (Chayn) Sun, Michael F. Worboys. "GIS - A Computing Perspective", CRC Press, 2023 Publication	<1 %
80	Mohammad Dehghan Rouzi, Myeounggon Lee, Jaewon Beom, Sanam Bidadi et al. "Quantitative biomechanical analysis in	<1 %

PLAGIARISM CHECK RESULT

validating a video-based model to remotely assess physical frailty: a potential solution to telehealth and globalized remote-patient monitoring", Biomedical Engineering Letters, 2024

Publication

81	Suryadev Singh, Shubham Kumar, Sandeep Kumar Singh. "chapter 10 Mobile App Testing and the AI Advantage in Mobile App Fine-Tuning", IGI Global, 2024	<1 %
Publication		
82	Submitted to Taylor's Education Group	<1 %
Student Paper		
83	Submitted to University College London	<1 %
Student Paper		
84	kodytechnolab.com	<1 %
Internet Source		
85	library.ndsu.edu	<1 %
Internet Source		
86	repository.kemu.ac.ke:8080	<1 %
Internet Source		
87	www.ijritcc.org	<1 %
Internet Source		

PLAGIARISM CHECK RESULT

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1 of 1



FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	Yuki Tan Lok Yee
ID Number(s)	20ACB05030
Programme / Course	FICT/CS
Title of Final Year Project	NutriLife: Empowering Health with Diet and Nutrition App

Similarity	Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)
Overall similarity index: <u>11</u> % Similarity by source Internet Sources: <u>6</u> % Publications: <u>7</u> % Student Papers: <u>6</u> %	
Number of individual sources listed of more than 3% similarity: -	
Parameters of originality required and limits approved by UTAR are as Follows: (i) Overall similarity index is 20% and below, and (ii) Matching of individual sources listed must be less than 3% each, and (iii) Matching texts in continuous block must not exceed 8 words <i>Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.</i>	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

Signature of Supervisor

Name: Ts Wong Chee Siang

Date: 13th September 2024

Signature of Co-Supervisor

Name: _____

Date: _____

Bachelor of Computer Science (Honours)
Faculty of Information and Communication Technology (Kampar Campus), UTAR



UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	20ACB05030
Student Name	YUKI TAN LOK YEE
Supervisor Name	TS WONG CHEE SIANG

TICK (✓)	DOCUMENT ITEMS
	Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
✓	Title Page
✓	Signed Report Status Declaration Form
✓	Signed FYP Thesis Submission Form
✓	Signed form of the Declaration of Originality
✓	Acknowledgement
✓	Abstract
✓	Table of Contents
✓	List of Figures (if applicable)
✓	List of Tables (if applicable)
✓	List of Symbols (if applicable)
✓	List of Abbreviations (if applicable)
✓	Chapters / Content
✓	Bibliography (or References)
✓	All references in bibliography are cited in the thesis, especially in the chapter of literature review
✓	Appendices (if applicable)
✓	Weekly Log
✓	Poster
✓	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)
✓	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report.

(Signature of Student)

Date: 13th September 2024