

QUETELET INDEX

INTERDISCIPLINARY PROJECT

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Technology in Information Technology

by

KAVIYA NIVASHINI V (Reg.No-41120091)



DEPARTMENT OF INFORMATION TECHNOLOGY

SCHOOL OF COMPUTING

SATHYABAMA

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

CATEGORY-1 UNIVERSITY BY UGC

Accredited "A++" by NAAC

12B Status by UGC |Approved by AICTE

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI-600119

MAY 2024



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited "A++" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF INFORMATION TECHNOLOGY

BONAFIDE CERTIFICATE

This is to certify that this internship Report is the Bonafide work of **Kaviya Nivashini V (41120091)** who carried out the project entitled "**Quetelet Index**" under my supervision from JANUARY 2024 to MAY 2024.

Internal Guide

Dr.C.Geetha,M.Tech.,Ph.D.,

Head of the Department

Dr.R. Subhashini, M.E., Ph.D.

Submitted for Viva voce Examination held on

8/5/24

Internal Examiner
External Examiner

DECLARATION

I, Kaviya Nivashini V (41120091) hereby declare that the Project Report entitled “**Quetelet Index**” done by me under the guidance of **Dr.C.Geetha,M.Tech.,Ph.D.**, and is submitted in partial fulfilment of the requirements for the award of Bachelor of Technology degree in Information Technology.

DATE: 8/5/24

PLACE: CHENNAI

V-Kaviya Nivashini

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

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

I convey my thanks to **DR. T. Sasikala M.E., PH.D., Dean**, School of Computing, **Dr.R.Subhashini M.E., Ph.D.**, Head of the Department of Information Technology for providing me necessary support and details at the right time during the progressive reviews.

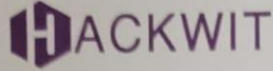
I would like to express my sincere and deep sense of gratitude to my Project Guide **Ms. Dr.C.Geetha,M.Tech.,Ph.D.**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

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

CERTIFICATE OF COMPLETION



CERTIFICATE OF COMPLETION



HACKWIT TECHNOLOGIES PVT LTD
Redefine by Technologies

149, 1C, 2nd Floor, Mount Poonamallee Rd,
Ramapuram, Chennai - 600089.
admin@hackwittechnologies.com
www.hackwittechnologies.com

10.02.2024

CERTIFICATE FOR INTERNSHIP

TO WHOMSOEVER IT MAY CONCERN

This is to Certify that **Kaviya Nivashini V (Reg No: 41120091)** student of Sathyabama Institute of Science and Technology, B. Tech Information Technology, has successfully completed the Internship at Hackwit Technologies from 22nd January to 30th January 2024.

During the internship, **Kaviya Nivashini V** has closely worked as a part of the **Flutter** by carrying out the project entitled "**BMI Calculator**".

She demonstrated good developing and learning skill with self-motivated attitude to learn new things. Her contribution to implement the Flutter was good. We wish her all the best for her future endeavors.

Best Regards,

A handwritten signature in black ink, appearing to be 'Ravikumar S'.

Ravikumar S
CEO & Managing Director
Hackwit Technologies Pvt Ltd

ABSTRACT

The Quetelet Index is a simple yet widely used tool in assessing an individual's body composition and potential health risks associated with weight. It provides a numerical value derived from a person's weight and height, offering a rough estimate of their body fat percentage. This calculation serves as a general indicator of whether an individual falls within a healthy weight range or if they may be at risk of health issues due to being underweight, overweight, or obese. The formula for calculating Quetelet Index is relatively straightforward: it involves dividing an individual's weight in kilograms by the square of their height in meters. The resulting number categorizes individuals into different ranges, each associated with varying degrees of body fat and potential health implications. While the calculation is a valuable initial screening tool, it's important to recognize its limitations. Quetelet Index does not directly measure body fat percentage or account for factors such as muscle mass, bone density, or distribution of fat throughout the body. Therefore, individuals with higher muscle mass, such as athletes or bodybuilders, may have a higher despite having a healthy body composition. Moreover, it does not differentiate between different types of body fat, such as visceral fat (fat around the organs) and subcutaneous fat (fat under the skin). Visceral fat, in particular, is associated with increased health risks, including cardiovascular disease and type 2 diabetes, even in individuals with a normal Quetelet Index. Therefore, relying solely on to assess health risks may not provide a comprehensive picture of an individual's overall health status. Despite its limitations, it remains a widely used tool due to its simplicity and accessibility. It can help healthcare professionals quickly identify individuals who may be at risk of obesity-related health issues and guide further evaluation and interventions.

TABLE OF THE CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	IV
	LIST OF ABBREVIATIONS	X
	LIST OF FIGURES	IX
1	INTRODUCTION	1
	1.1 Key Objectives	2
	1.2 System Overview	3
	1.3 Features and Functionality	4
2	METHODOLOGY	6
	2.1 Aim	6
	2.2 Scope	6
	2.3 Material used	8
	2.4 Methods used	10
	2.5 System Architecture	11
	2.6 Algorithm used	14
3	RESULT AND DISCUSSION	15

4	CONCLUSION AND FUTURE WORK	22
4.1	Conclusion	22
4.2	Future Work	24
	REFERENCES	26
	APPENDICES	27
A.	Sample Code	27
B.	Screen Shots	34

LIST OF FIGURES

FIGURE NO.	NAME OF THE FIGURE PAGE	PAGE NO.
2.1	System Architecture	11
3.1	Login Page Interface	18
3.2	Firebase Configurations	19
3.3	Initializing Firebase	20

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
UI	- User Interface
UX	- User Experience
iOS	- iPhone Operating System
UAE	- User Acceptance Testing
IDE	- Integrated Development Environment

CHAPTER 1

1. INTRODUCTION

The overweight (Quetelet Index for age and sex from the 85th percentile to the 95th percentile) and obesity (Quetelet Index I for age and sex greater than or adequate to the 95th percentile) are related to the present and future health of youth also as their academic success. Youth classified as overweight or obese are more likely to be diagnosed with prediabetes (impaired fasting glucose or impaired glucose tolerance) and have more cardio metabolic risk factors than normal weight youth. Various methods for assessing body composition, used for children and adolescents in research settings (for example, hydrodensitometry, plethysmography of air displacement, isotope dilution, two-energy X-ray absorptiometry). The use of these methods in a community setting is limited. As a result, measuring height and weight to calculate Quetelet Index is recommended for overweight/obese people in screening for the younger generation. $(\text{body weight (kg)} / \text{height (square meters)})$. Android is used in over 190 countries and powers hundreds of millions of mobile devices. It has the largest installed base of any mobile platform and is rapidly expanding—every day, a million new Android users turn on their devices for the first time and begin searching for apps, games, and other digital content. Android provides you with a world-class platform for developing apps and games for Android users around the world, as well as an open marketplace for quickly distributing them. This Android application has handy tools for arithmetic, scientific, and converting calculations. It would be particularly useful for students, as they account for the bulk of smartphone users in today's world, and they require tools to help them with the lengthy calculations they must perform in their studies.

1.1 Key Objectives

Creating a Quetelet Index calculator in Flutter involves several key objectives:

1.User Interface Design: Designing an intuitive and user-friendly interface for inputting height and weight values.

2.Data Input Handling: Implementing mechanisms to accept user input for height and weight, potentially including validation to ensure the input is within appropriate ranges.

3.Quetelet Index Calculation Logic: Developing the logic to calculate the Quetelet Index based on the provided height and weight inputs. This involves applying the Quetelet Index formula: $\text{Quetelet Index} = \text{weight (kg)} / (\text{height (m)})^2$.

4.Displaying Results: Presenting the calculated Quetelet Index value to the user in a clear and understandable format, along with an interpretation of what the Quetelet Index value signifies in terms of health.

5.Styling and Theming: Applying appropriate styling and theming to ensure consistency and aesthetics throughout the application.

6.Error Handling: Implementing error handling mechanisms to manage potential errors or edge cases, such as dividing by zero or invalid input.

7.Unit Conversion: Optionally, providing functionality to allow users to input height and weight in different units (e.g., feet/inches and pounds, or centimeters and kilograms).

8.Testing: Conducting thorough testing to ensure the calculator functions correctly under various scenarios and edge cases.

By addressing these objectives, you can create an effective Quetelet Index calculator application in Flutter.

1.2 System Overview

In a BMI (Body Mass Index) calculator developed using Flutter, the system overview typically involves several key components:

1.User Interface(U):

- The UI is developed using Flutter widgets to create a user-friendly interface.
- It includes input fields for the user to enter their height and weight.
- Buttons are provided for the user to submit their data and calculate their BMI.

2. Business Logic Layer:

- This layer handles the calculation of BMI based on the user's input.
- It may include functions to validate user input and perform the BMI calculation using the formula: $BMI = \text{weight (kg)} / (\text{height (m)} * \text{height (m)})$.

3.Data Handling:

- Processes and manages user input data with validation to ensure correctness.
- Convert input values between metric and imperial units if necessary.

4.Quetelet Index Calculation:

- Validates user input and calculates the Quetelet Index using provided data.

5.Output Display:

- Displays the calculated Quetelet Index value to the user clearly.
- Additional information, such as categorization based on the Quetelet Index, may be provided with associated health implications.

6.Integration with Flutter Framework:

- Integrated into the Flutter framework, utilizing its features and functionalities for building cross-platform mobile applications.
- Utilizes Flutter widgets for UI components and Flutter's reactive framework for updating UI based on user input and calculated results.

9.Testing and Debugging:

- Undergoes testing to ensure functionality and reliability.
- Unit tests verify correctness of components, including the Quetelet Index calculation algorithm.

1.3 Features and Functionality:

To create a BMI calculator app in Flutter, you'll need to define both the features and functionality. Here's a basic outline:

1.3.1 Features:

- Calculate Quetelet Index: Allow users to input their height and weight, then calculate their Quetelet Index using the formula:
$$\text{Quetelet Index} = \text{weight(kg)} / (\text{height(m)} * \text{height(m)}).$$
- QueteletIndex Categories: Display the user's categor(Underweight,

Normal weight, Overweight, or Obese) based on the calculated Index value.

- Metric/Imperial Units: Provide an option for users to input their height and weight in either metric (centimeters/kilograms) or imperial (inches/pounds) units.
- Interactive UI: Design a user-friendly interface with input fields, buttons, and clear labels to guide users through the process.
- Feedback: Provide feedback to the user about their Quetelet Index category, possibly with color-coded indicators or descriptive text.
- Reset: Include a button to reset the input fields and clear the calculated result.

1.3.2 Functionality:

- Input Validation: Ensure that the user cannot input negative values or leave the input fields blank.
- Unit Conversion: Convert the user's input between metric and imperial units as necessary for calculations.
- Index Calculation: Implement the Quetelet Index formula to calculate the Index based on the user's input.
- Quetelet Index Category Determination: Based on the calculated Index value, determine the corresponding Quetelet Index category and display it to the user.
- Responsive Design: Make sure the app layout is responsive to different screen sizes and orientations.
- Localization: Support multiple languages if needed, especially for units of measurement and Quetelet Index category labels. Implementing these features and functionality will create a comprehensive BMI calculator app in Flutter.

CHAPTER -2

METHODOLOGY

2.1 Aim:

The aim of this project is to create where people can easily target their weight easily efficiently with each other and provide ease and comfort to the people. This is an one handed application users can carried out by this way is as simple. Doesn't need any multiple or mathematical calculation. The feature aims to eliminate paper based calculations or some weighing machines. Users can check their body mass at any time and any places with this single application. This online application is very simple to calculate by just entering their weight and height.

2.2 Scope:

The scope of a Quetelet Index application developed using Flutter encompasses various aspects, including its functionality, features, target audience, and potential extensions. Here's a brief overview:

1. Functionality:

- The primary functionality of the Quetelet Index is to calculate the Quetelet Index based on user input for height and weight.
- It categorizes the calculated Index into predefined categories such as Underweight, Normal weight, Overweight, or Obese.
- It provides users with the option to input their data in either metric (centimeters/kilograms) or imperial (inches/pounds) units.
- The application should offer a reset option to clear input fields and recalculated Quetelet Index results.

2. Features:

- **Interactive UI:** Design a user-friendly interface with intuitive input fields, buttons, and clear labels to guide users through the process.
- **Feedback:** Provide feedback to users about their Quetelet Index category, potentially using color-coded indicators or descriptive text.
- **Metric/Imperial Units:** Implement functionality to allow users to toggle between metric and imperial units seamlessly.
- **Reset:** Include a button to reset input fields and clear the calculated Quetelet Index result for convenience.

3. Target Audience:

- The target audience for the Quetelet Index calculator includes individuals interested in monitoring their weight and overall health.
- It can cater to a wide range of users, including health-conscious individuals, fitness enthusiasts, and healthcare professionals.

4. Extensions:

- **User Profiles:** Allow users to create profiles to track their Quetelet Index over time and set goals.
- **Integration with Health Data:** Integrate with platforms or APIs to sync data with health tracking apps or devices.
- **Additional Health Metrics:** Expand the application to calculate and track other health metrics such as body fat percentage, waist-to-hip ratio, etc.
- **Educational Resources:** Provide educational resources or links to articles about healthy weight management and lifestyle choices.
- **Interactive UI:** Design a user-friendly interface with intuitive input fields, buttons, and clear labels to guide users through the process.

2.3 Material used:

When building a a Quetelet Index using Flutter, a popular framework for building cross-platform mobile applications, developers utilize various materials to create a functional and visually appealing app. Here's a brief overview:

- **Flutter Framework:** Flutter provides the foundation for building the app. It uses the Dart programming language and offers a rich set of UI components, known as widgets, for creating the user interface.
- **Widgets:** Flutter's widget-based approach allows developers to create custom UI components and arrange them to design the app's interface. Widgets like Text, TextField, Button, and Slider are commonly used in a a Quetelet Index to display text, input weight and height, and provide interactive elements for calculations.
- **Mathematical Calculations:** Developers implement the calculation formula within the Dart codebase. This involves taking user input for weight and height, applying the a Quetelet Index formula ($\text{a Quetelet Index} = \text{weight (kg)} / (\text{height (m)})^2$), and displaying the calculated Index result.
- **State Management:** Flutter provides various options for managing the app's state, such as StatefulWidget and Provider, to handle changes in user input and update the UI accordingly. State management ensures that the calculator reflects real-time changes as users input their weight and height.
- **Validation and Error Handling:** Developers include logic to validate user input, ensuring that weight and height values are within reasonable ranges and handling any potential errors gracefully.

- **Styling and Theming:** Flutter allows developers to style and theme the app to enhance its visual appeal and provide a consistent user experience. Developers can customize colors, fonts, and layout to create an attractive and user-friendly interface.
- **Testing Tools:** Flutter offers testing tools like Flutter Test and integration with popular testing frameworks such as Mockito for unit testing and Flutter Driver for widget testing. These tools help ensure the reliability and functionality of the a Quetelet Index across different devices and screen sizes.
- **Deployment Platforms:** Once the a Quetelet Index is developed, it can be deployed to various platforms, including iOS and Android, using Flutter's platform-specific deployment tools. Developers may also utilize platforms like Google Play Store and Apple App Store to distribute the app to users.
- **State Management:** Flutter provides various options for managing the app's state, such as StatefulWidget and Provider, to handle changes in user input and update the UI accordingly. State management ensures that the calculator reflects real-time changes as users input their weight and height.

In summary, building a Quetelet Index using Flutter involves leveraging the framework's widgets, state management capabilities, mathematical calculations, validation and error handling, styling and theming options, testing tools, and deployment platforms to create a fully functional and visually appealing app for estimating a Quetelet Index.

2.4 Methods used:

Here's a brief overview of methods commonly used in a Quetelet Index implemented using Flutter:

- **Calculate Method:** This method calculates using the formula: $\text{Quetelet Index} = \text{weight (kg)} / (\text{height (m)})^2$. It takes the weight and height as input parameters, performs the calculation, and returns the value.
- **Category Method:** This method categorizes the calculated into different categories such as underweight, normal weight, overweight, and obesity based on predefined ranges. It takes the Index value as input and returns the corresponding category.
- **Input Validation Methods:** These methods ensure that the input provided by the user (weight and height) is valid before performing the calculation. They check if the input values are within acceptable ranges and display appropriate error messages if invalid input is detected.
- **UI Update Methods:** These methods update the user interface to display the calculated Index value, category, and any error messages. They typically update the relevant widgets such as text fields, labels, or alert dialogs with the computed information.
- **Reset Method:** This method resets the input fields and clears any displayed Quetelet Index information or error messages, allowing the user to input new values and recalculate if needed.

- **Conversion Methods:** If the input fields accept measurements in different units (e.g., kilograms and centimeters or pounds and inches), conversion methods may be needed to convert between these units before performing the calculation.
- **Event Handling Methods:** These methods handle user interactions such as button clicks or text input changes. They trigger the appropriate actions such as validating input, calculating Index value, or resetting the input fields based on user actions. These methods would be implemented within the Flutter framework using Dart programming language to create a Quetelet Index app with a user-friendly interface and accurate calculations.
- **Deployment Platforms:** Once the a Quetelet Index is developed, it can be deployed to various platforms, including iOS and Android, using Flutter's platform-specific deployment tools.

2.5. System Architecture:

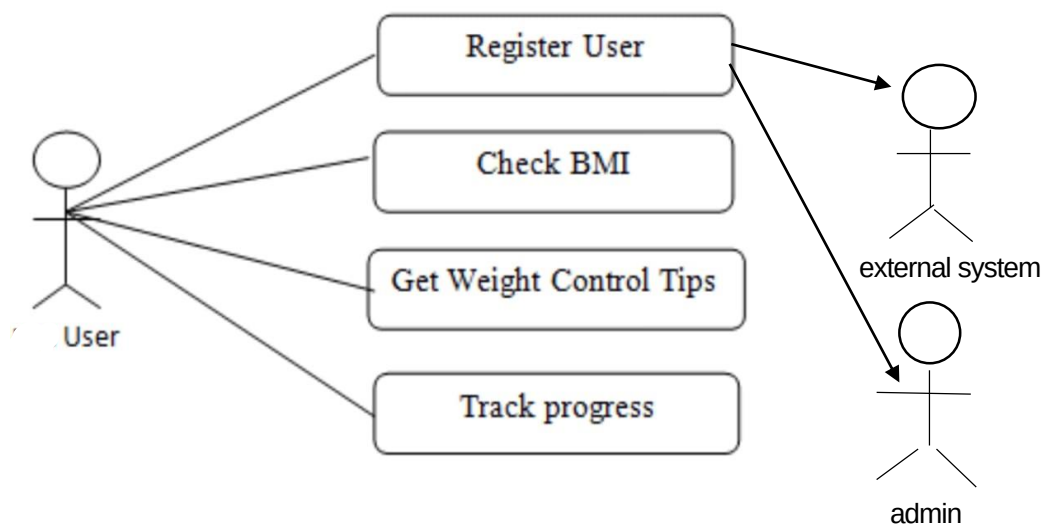


Fig 2.1: System Architecture

Actor: User:

- Calculate Index: This represents the primary functionality of the Quetelet Index where users input their height and weight to compute their Index.
- View Result: After entering height and weight, users can view the calculated value.
- Interpret: Provides users with an interpretation of their Index value, indicating whether it's underweight, normal weight, overweight, or obese.
- Save Record: Users may have the option to save their records for future reference or tracking purposes.

Actor: Admin:

- Manage Users: If the system includes user accounts, an admin might have the capability to manage user accounts, including adding, deleting, or modifying user information.

Actor: External System:

- Data Storage: Interaction with an external system for storing user data, including Quetelet Index records, if the application incorporates cloud storage or a database.

2.4.2 Design:

In developing a Quetelet Index Application using Flutter, several design considerations are important for creating an intuitive and user-friendly application. Here are some key design elements commonly used in such applications:

- **Responsive Design:** Ensure that the UI layout adjusts seamlessly across different screen sizes and orientations, providing a consistent user experience on various devices.
- **User Input Validation:** Implement validation mechanisms to ensure that user input for height and weight is within valid ranges and of appropriate data types. This helps prevent calculation errors and enhances user experience by providing immediate feedback on incorrect inputs.
- **Interpretation of Results:** Include features to interpret the result and provide users with understandable feedback on their weight status (e.g., underweight, normal weight, overweight, or obese). This can be achieved through text descriptions or visual indicators, such as color-coded categories.
- **Accessibility:** Ensure accessibility features are incorporated into the app, such as support for screen readers, scalable text, and high contrast modes. This ensures that the app is usable by individuals with diverse abilities.
- **Localization:** Support multiple languages and regions to cater to a global audience. Allow users to switch between different language options to improve accessibility and usability for non-English speakers.
- **Performance Optimization:** Optimize app performance by minimizing unnecessary computations and ensuring smooth rendering of UI elements. Consider lazy loading techniques for resource-intensive operations to enhance app responsiveness.

2.5 Algorithm used:

- Input Collection:

- 1) The user inputs their weight (in kilograms or pounds) and height (in meters or inches).
- 2) Some calculators may also ask for additional information such as age and gender, although these are not directly used in the calculation itself.

- Conversion (if necessary):

- 1) If the user inputs weight in pounds and/or height in inches, conversion to kilograms and/or meters respectively may be required for the formula.

- Calculation:

- 1) Quetelet Index is calculated using the following formula:

$$\textbf{Quetelet Index} = \text{height}^2 \text{weight}$$

- 2) Where weight is in kilograms and height is in meters (or weight in pounds and height in inches if using the imperial system).

- Interpretation:

- 1) Based on the calculated value, the user's weight is interpreted. Common categories include underweight, normal weight, overweight, and obese. The interpretation may be accompanied by additional information or recommendations regarding health and fitness.

CHAPTER-3

RESULT AND DISCUSSION

RESULT:

1) Login Functionality:

- Users can create an account or log in with existing credentials.
- The application verifies the credentials and grants access to the quiz interface upon successful login.

2) Adding Credentials:

- Users can sign up by providing necessary information such as username, email, and password.
- The application should ensure secure storage of user credentials, preferably by hashing passwords and storing them in a database.

3) Calculate your Quetelet Index:

- Enter the necessary information that asked by the application to get the Quetelet Index value.
- After the process click the Calculate button that given below.

4) About:

“Quetelet Index is that the metric currently in use for outlining anthropometric height and weight characteristics in adults and for classifying them into groups. The common interpretation is that it represents an index of an individual's fatness. It is also widely used as a risk factor for the event of or the prevalence of several health issues. The Quetelet Index Calculator App is a software program that eliminates the need for more manual hours to calculate and locate it for a specific person with a single click. This app provides all of the information in both standards that is not available in any other app. The major goal is to keep one's health in good shape.

The Quetelet Index App provides us with all of the necessary information, such as health recommendations and advice on what to eat and what to avoid.”

5) result

A Quetelet Index provides users with their Body Mass Index based on their weight and height inputs. The result serves as an indicator of a person's body fatness and can help assess their overall health status. Here's a brief summary of what the result signifies:

a) Underweight:

- less than 18.5
- Indicates that the individual may be underweight, which could be associated with health risks such as nutrient deficiencies or weakened immunity.

b) Normal Weight:

- between 18.5 and 24.9
- Suggests that the individual has a healthy weight relative to their height, which is associated with a lower risk of chronic diseases and overall good health.

c) Overweight:

- between 25 and 29.9
- Indicates that the individual may be overweight, which can increase the risk of developing various health issues such as heart disease, diabetes, and high blood pressure.

d) Obesity:

- 30 or greater
- Signifies that the individual falls into the obese category, indicating a significantly higher risk of obesity-related health problems, including cardiovascular disease, stroke, and certain types of cancer.

Quetelet Index should be interpreted in conjunction with other factors such as waist circumference, body fat percentage, and overall health status for a comprehensive evaluation. It's important to note that while Quetelet Index is a useful screening tool, it has limitations and may not accurately assess body composition in individuals with high muscle mass or certain medical conditions. Therefore, Quetelet Index should be interpreted in conjunction with other factors such as waist circumference, body fat percentage, and overall health status for a comprehensive evaluation. Additionally, categories may have different implications for children and adolescents due to differences in growth patterns and development. After the calculation, the result is sent back to the UI layer for presentation to the user. This may include displaying the value, interpreting its meaning in terms of health categories. It is also widely used as a risk factor for the event of or the prevalence of several health issues. The Quetelet Index Calculator App is a software program that eliminates the need for more manual hours to calculate and locate it for a specific person with a single click. This app provides all of the information in both standards that is not available in any other app. The major goal is to keep one's health in good shape.

3.1.2 Dart Codes:

i) `main.dart`: This file is typically the entry point of a Flutter application. It initializes the app, sets up necessary configurations, and runs the main application widget.

ii) `loginPage.dart`: This file likely contains the UI and logic for user login. It includes form fields for entering credentials and a mechanism to authenticate users.



Fig:3.1: Login Page Interface

iii) QueteletIndexPage.dart: This file represents the main content of your app after a successful login or signup. You can calculate the Quetelet Index by entering the necessary data that are asked.

iv) AboutPage.dart: It helps to user can verify the about page if user have any queries or and helps to get to know about the application.

3.1.3 FIREBASE:

i)Create a Firebase Project:

- Go to the Firebase Console.
- Click on "Add Project" and follow the setup instructions.

ii)Configure Flutter Project:

- Add the Firebase configuration files to your Flutter project.
- Follow the instructions provided in the Firebase Console to add google-services.json for Android and GoogleService-Info.plist for iOS.
- These files contain configuration details needed for your app to connect to Firebase services.

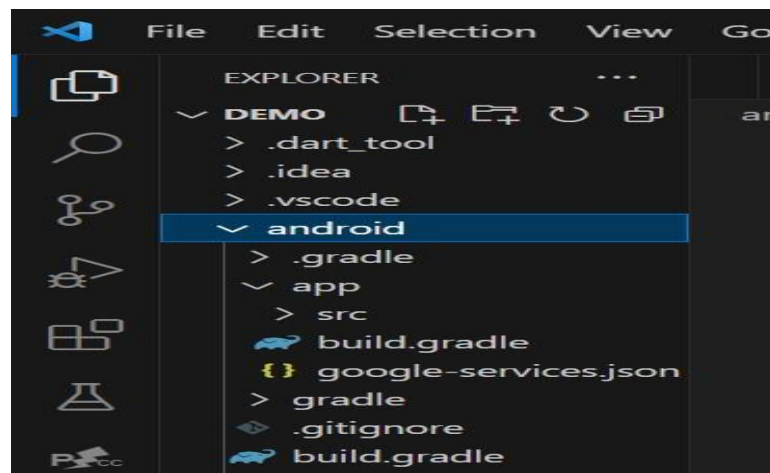


Fig:3.2: Firebase Configurations

iii)Add Firebase Dependencies:

Open the pubspec.yaml file and add the following dependencies: yamll
code: dependencies:

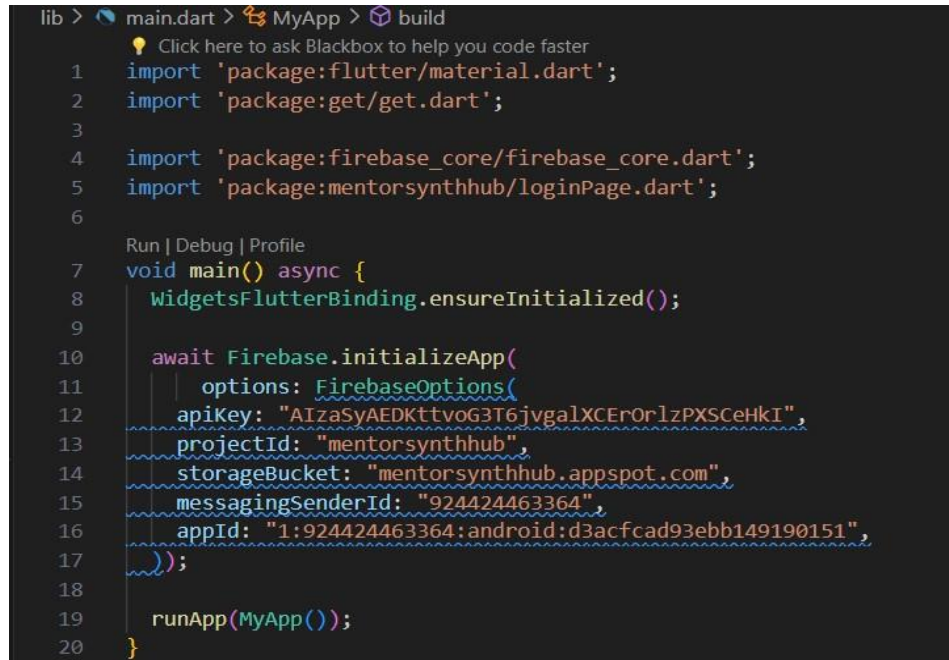
- firebase_core: ^2.25.3
- firebase_auth: ^4.17.5

iv)Initialize Firebase:

In your main.dart or wherever you initialize your Flutter app, make sure to initialize Firebase. If you want to implement user authentication,

Firebase provides the `firebase_auth` package. You can use it to sign in users using various methods like email/password, Google Sign-In, etc.

Dart code:



```
lib > main.dart > MyApp > build
Click here to ask Blackbox to help you code faster
1 import 'package:flutter/material.dart';
2 import 'package:get/get.dart';
3
4 import 'package:firebase_core/firebase_core.dart';
5 import 'package:mentorsynthhub/loginPage.dart';
6
7 void main() async {
8   WidgetsFlutterBinding.ensureInitialized();
9
10  await Firebase.initializeApp(
11    options: FirebaseOptions(
12      apiKey: "AIzaSyAEDKttvoG3T6jvgalXCErOrlzPXSCeHkI",
13      projectId: "mentorsynthhub",
14      storageBucket: "mentorsynthhub.appspot.com",
15      messagingSenderId: "924424463364",
16      appId: "1:924424463364:android:d3acfcad93ebb149190151",
17    ));
18
19  runApp(MyApp());
20 }
```

Fig:3.3: Initializing Firebase

3.2 DISCUSSIONS:

Creating a discussion feature in a Quetelet Index using Java involves several steps:

1. Database Setup:

- Choose a database system (e.g., MySQL, PostgreSQL) to store discussion
- user ID, Nmae, content, timestamp, etc.

2. Backend Development:

- Set up a Java backend server using a framework like Spring Boot.
- Create API endpoints for discussions, including functionalities like creating, reading, updating, and deleting posts.

- Implement authentication and authorization to ensure that only authorized users can access the discussion features.

3. User Interface:

- Creating a user interface (UI) using Flutter involves designing a visually appealing layout that allows users to input their height and weight easily and view the calculated result.

4. Integration with Calculation:

- Incorporating Quetelet Index functionality into a Flutter application presents an opportunity to provide users with a comprehensive health monitoring tool.
- By seamlessly integrating Quetelet Index, developers can empower users to assess their body composition and make informed decisions regarding their fitness goals.

5. Security:

- Implement security measures to prevent common vulnerabilities like SQL injection and Cross-Site Scripting (XSS).
- Ensure user data is securely stored and transmitted.

6. Testing:

- Thoroughly test the discussion feature to ensure it works correctly and is free of bugs.

CHAPTER 4

CONCLUSION AND FUTURE WORK

4.1 CONCLUSION:

In the realm of health and fitness, the Quetelet Index stands as a fundamental metric, serving as a quick indicator of one's overall body composition and potential health risks associated with weight. Integrating calculation into mobile applications has become increasingly popular, aligning with the modern lifestyle where individuals seek instant access to health-related information on their smartphones. In this context, leveraging the Flutter framework for developing a Quetelet Index proves to be a pragmatic choice, offering a blend of efficiency, versatility, and user-centric design.

Flutter, with its cross-platform capabilities and rich set of widgets, emerges as an optimal framework for crafting Quetelet Index that are both aesthetically pleasing and highly functional. The flexibility of Flutter allows developers to create fluid and responsive user interfaces, ensuring an intuitive user experience across various devices and screen sizes. Whether it's designing a sleek interface with customizable themes or implementing interactive elements for user input, Flutter provides developers with the tools needed to bring their Quetelet Index visions to life.

Moreover, the robustness of Flutter facilitates seamless integration with backend services for data processing and storage. By leveraging Flutter's compatibility with popular programming languages like Dart, developers can implement Quetelet Index algorithms with precision and

efficiency. Whether it's utilizing mathematical formulas to compute or incorporating additional features such as tracking progress over time, Flutter empowers developers to create Quetelet Index that are comprehensive and feature-rich.

Furthermore, Flutter's support for third-party plugins and packages enhances the capabilities of Quetelet Index, allowing developers to incorporate additional functionalities such as data visualization, health data synchronization, and social sharing. These integrations enable Quetelet Index apps to transcend basic functionality, providing users with a holistic approach to managing their health and wellness goals.

From a user perspective, Flutter-based Quetelet Index offer several advantages. The intuitive and visually appealing interfaces ensure ease of use, allowing users to input their height and weight effortlessly and obtain instant results. Additionally, features such as personalized recommendations based on categories, trends analysis, and goal setting enhance user engagement and motivation, fostering a sense of empowerment and accountability in managing one's health.

In conclusion, developing a Quetelet Index using Flutter represents a convergence of technology and wellness, catering to the growing demand for accessible and user-friendly health management tools in the digital age. By harnessing the capabilities of Flutter, developers can create Quetelet Index that not only deliver accurate results but also inspire users to make informed decisions about their health and well-being. As mobile technology continues to evolve, Flutter stands poised to remain a cornerstone framework for building innovative and impactful health and fitness applications, shaping a healthier future for individuals worldwide.

4.2. FUTURE WORK:

Future work on Quetelet Index could focus on enhancing user engagement, personalization, and integration with emerging technologies. Here's a brief overview:

Enhanced User Engagement: Future Quetelet Index could incorporate gamification elements to make the experience more interactive and enjoyable. Features like achievement badges, challenges, and rewards for achieving health goals could motivate users to track their Index regularly and maintain healthy habits.

Personalization: Implementing machine learning algorithms to provide personalized recommendations based on users' Quetelet Index, fitness goals, and health history could significantly improve the effectiveness of Quetelet Index calculators. Tailored advice on nutrition, exercise routines, and lifestyle modifications could empower users to make sustainable changes to improve their health.

Integration with Wearable Devices: Integrating Quetelet Index with wearable devices such as smartwatches and fitness trackers would enable seamless tracking of physical activity, sleep patterns, and other health metrics. This integration would provide users with a comprehensive view of their health status and progress towards their fitness goals.

Health Data Analytics: Utilizing big data analytics techniques to analyze aggregated user data from calculators could uncover valuable insights into population health trends, risk factors for chronic diseases, and effectiveness of different interventions. This data-driven approach

could inform public health initiatives and healthcare policies aimed at combating obesity and promoting healthy lifestyles.

Accessibility and Inclusivity: Ensuring that Quetelet Index calculators are accessible to diverse user populations, including individuals with disabilities or those with limited access to healthcare resources, is crucial. Future work could focus on improving the accessibility features of Quetelet Index apps, such as voice commands, screen reader compatibility, and multilingual support.

By focusing on these areas of future work, Quetelet Index can evolve into more sophisticated and user-centric tools for promoting health and wellness in the digital age.

REFERENCE

- Cummings, M., 2004. Automation bias in intelligent time critical decision support systems, in: AIAA 1st Intelligent Systems Technical Conference. p. 6313
- Folstad, Asbjorn (2018), INTERACTION WITH AI – MODULE 2 - Session 1, UIO Retrieved from <https://www.uio.no/studier/emner/matnat/ifi/IN5480/h18/undervisningsmateriale/interaction-with-ai—module-2---session-1---v02.pdf>
- Hung, V., Gonzalez, A., & DeMara, R. (2009, February). Towards a context- based dialog management layer for expert systems. In Information, Process, and Knowledge Management, 2009. eKNOW'09. International Conference on (pp. 60-65). IEEE.
- Jung, M., Hinds, P., 2018. Robots in the Wild: A Time for More Robust Theories of Human-Robot Interaction. ACM Trans. Hum.-Robot Interact. 7, 2:1–2:5.

APPENDICES

A.SAMPLE CODE

i) Main Code:

```
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
import 'package:get/get.dart'; // Import Get package
import 'package:learnle/customs/translator_page.dart';
import 'package:learnle/firebase_options.dart';
import 'package:learnle/models/bottom_navigation_bar.dart';
import 'package:learnle/pages/homePage.dart';
import 'package:learnle/screens/login_screen.dart';
import 'package:firebase_auth/firebase_auth.dart';

void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp();
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return GetMaterialApp(
      // Replace MaterialApp with GetMaterialApp
      debugShowCheckedModeBanner: false,
      title: 'LearnLE',
      theme: ThemeData(
        backgroundColor: Colors.white,
        appBarTheme: AppBarTheme(color: Colors.white, elevation: 0),
        visualDensity: VisualDensity.adaptivePlatformDensity,
      ),
      home: LoginScreen(),
    );
  }
}
```

ii)login_screen.dart

```
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
//import 'package:flutter_project/myHomePage.dart';
//import 'package:flutter_project/nextPage.dart';
import 'package:get/get.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:learnle/screens/home_screen.dart';
//import 'package:emiapp/MainPage.dart';

//import 'package:flutter_project/nextPage.dart';
final _firebase = FirebaseAuth.instance;

class LoginScreen extends StatefulWidget {
  const LoginScreen({super.key});

  @override
  State<LoginScreen> createState() => _LoginScreenState();
}

class _LoginScreenState extends State<LoginScreen> {
  final _formKey = GlobalKey<FormState>();
  void register() {}
  var _isLogin = true;
  var _enteredEmail = "";
  var _enteredPassword = "";
  void _submit() async {
    final isValid = _formKey.currentState!.validate();
    if (!isValid) {
      return;
    }
    _formKey.currentState!.save();
    try {
      if (_isLogin) {
        final userCredentials = await _firebase.signInWithEmailAndPassword(
          email: _enteredEmail, password: _enteredPassword);
```

```

        print(userCredentials);
        if (userCredentials.user != null) {
            Get.to(HomeScreen());
        }
    } else {
        final userCredentials = await _firebase.createUserWithEmailAndPassword(
            email: _enteredEmail, password: _enteredPassword);
        print(userCredentials);
    }
} on FirebaseAuthException catch (error) {
    if (error.code == 'email_already_in_use') {}
    ScaffoldMessenger.of(context).clearSnackBars();
    ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(
            content: Text(error.message ?? 'Authentication failed'),
        ),
    );
}
}

@override
Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(
            leading: GestureDetector(
                onTap: () {
                    Get.to(HomeScreen());
                },
                child: Icon(Icons.arrow_back)),
            title: Text(
                '      LOGIN',
                style: TextStyle(color: Colors.white),
            ),
            backgroundColor: Colors.blue,
        ),
        backgroundColor: Colors.white,
        body: Center(
            child: SingleChildScrollView(

```

```
scrollDirection: Axis.vertical,
child: Column(mainAxisAlignment: MainAxisAlignment.center, children: [
  SizedBox(
    height: 35,
  ),
  Text(
    "Quetelet",
    style: TextStyle(
      color: Colors.black,
      fontWeight: FontWeight.bold,
      fontSize: 60.0),
  ),
  SizedBox(
    height: 0,
  ),
  SizedBox(
    height: 0,
  ),
  SizedBox(
    height: 5,
  ),
  Card(
    child: Form(
      key: _formKey,
      child: Column(
        children: [
          TextFormField(
            decoration: const InputDecoration(
              icon: Icon(Icons.email), labelText: 'Email'),
            keyboardType: TextInputType.emailAddress,
            autocorrect: false,
            textCapitalization: TextCapitalization.none,
            validator: (value) {
              if (value == null ||
                  value.trim().isEmpty ||
                  !value.contains('@')) {
                return 'Enter a valid email address';
              }
            }
          )
        ]
      )
    )
  )
]
```

```

        return null;
    },
    onSave: (value) {
        _enteredEmail = value!;
    },
),

//Icon(Icons.mail),
TextFormField(
    decoration: const InputDecoration(
        icon: Icon(Icons.password), labelText: 'Password'),
    obscureText: true,
    validator: (value) {
        if (value == null || value.trim().length < 6) {
            return 'Enter a valid password';
        }
        return null;
    },
    onSave: (value) {
        _enteredPassword = value!;
    },
),
ElevatedButton(
    onPressed: _submit,
    style: ElevatedButton.styleFrom(
        foregroundColor: Colors.black,
        backgroundColor: Color.fromARGB(
            255, 49, 93, 239), //maximumSize: Size.fromWidth(80)
    ),
    child: Text(
        _isLogin ? 'Login' : 'Signup',
        style: TextStyle(fontWeight: FontWeight.bold),
    ),
),
Padding(
    padding: const EdgeInsets.all(8.0),
    child: SizedBox(
        width: double.infinity,

```

[illegible]

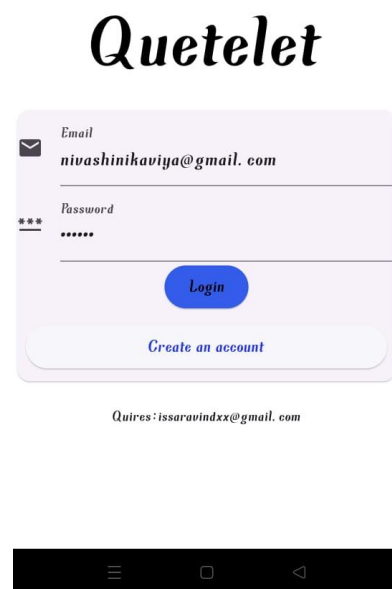
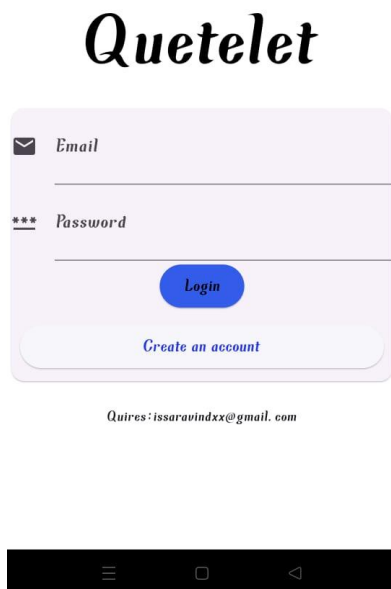
```
    ),  
    ),  
  );  
}  
}
```

iii) pubspec.yaml

```
name: learnle  
description: A new Flutter project.  
publish_to: "none"  
version: 1.0.0+1  
environment:  
  sdk: ">=2.19.6 <3.0.0"  
  
dependencies:  
  cupertino_icons: ^1.0.2  
  flutter:  
    sdk: flutter  
  get: ^4.6.6  
  
dev_dependencies:  
  cloud_firestore: ^4.5.2  
  firebase_auth: ^4.4.2  
  firebase_core: ^2.10.0  
  flutter_lints: ^3.0.2  
  flutter_svg: ^2.0.5  
  
  flutter_test:  
    sdk: flutter  
  flutter_tts: ^4.0.2  
  fluttertoast: ^8.2.1  
  translator: ^1.0.0  
  
flutter:  
  uses-material-design: true  
  - assets/logo1.png
```

B.SCREEN SHOTS

1)Login page:



2)sign up and password error:



Quetelet

Email
kaviya@gmail.com

Password

Enter a valid password

Signup

I already have a account

Quires:issaravindxx@gmail.com

Quetelet

Email

Password

Signup

I already have a account

Quires:issaravindxx@gmail.com



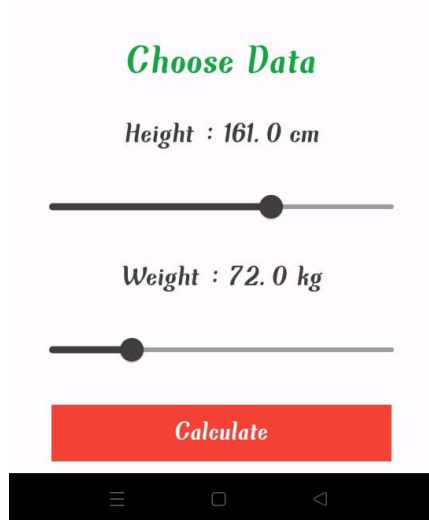
3)Quetelet Index:

The screenshot shows the Quetelet Index app interface. At the top, the title 'Quetelet Index' is displayed in white on a red background. Below the title, the calculated index value '22' is shown in large white digits. Underneath, the text 'Condition : Normal' is displayed in white. The lower section has a white background with the heading 'Choose Data' in green. It shows two input fields: 'Height : 161.0 cm' and 'Weight : 57.0 kg', each with a horizontal slider below it. At the bottom of this section is a red button labeled 'Calculate' in white. The entire app is running on an Android device, as indicated by the status bar at the top and the navigation bar at the bottom.

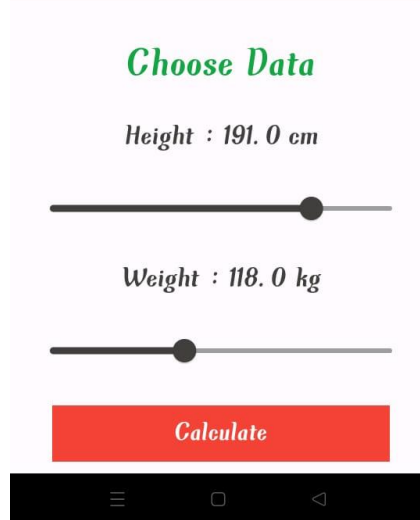
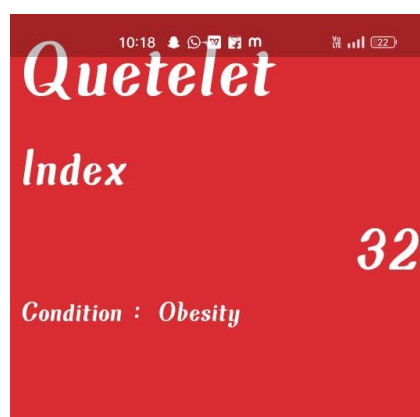
Normal

The screenshot shows the Quetelet Index app interface. At the top, the title 'Quetelet Index' is displayed in white on a red background. Below the title, the calculated index value '13' is shown in large white digits. Underneath, the text 'Condition : Underweight' is displayed in white. The lower section has a white background with the heading 'Choose Data' in green. It shows two input fields: 'Height : 140.0 cm' and 'Weight : 25.0 kg', each with a horizontal slider below it. At the bottom of this section is a red button labeled 'Calculate' in white. The entire app is running on an Android device, as indicated by the status bar at the top and the navigation bar at the bottom.

Underweight



Overweight



obesity