

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELAGAVI – 590 018



An Internship Project Report on

BMI Application

Submitted in partial fulfillment of the requirements for the VIII Semester of degree of
Bachelor of Engineering in Information Science and Engineering of Visvesvaraya
Technological University, Belagavi

by

Eshwar P V

1RN18IS043

Under the Guidance of

Mr. R Rajkumar

Associate Professor

Department of ISE



ESTD:2001

An Institute with a Difference

Department of Information Science and Engineering

RNS Institute of Technology

**Dr. Vishnuvaradhan Road, Rajarajeshwari Nagar post,
Channasandra, Bengaluru-560098**

2021-2022

RNS INSTITUTE OF TECHNOLOGY

Dr. Vishnuvaradhan Road, Rajarajeshwari Nagar post,
Channasandra, Bengaluru - 560098

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING



CERTIFICATE

Certified that the Internship work entitled **BMI Application** has been successfully completed by **Eshwar P V (1RN18IS043)** a bonafide student of **RNS Institute of Technology, Bengaluru** in partial fulfillment of the requirements of 8th semester for the award of degree in **Bachelor of Engineering in Information Science and Engineering of Visvesvaraya Technological University, Belagavi** during academic year **2021-2022**. The internship report has been approved as it satisfies the academic requirements in respect of internship work for the said degree.

Mr. R Rajkumar
Internship Guide
Associate Professor
Department of ISE

Dr. Suresh L
Professor and HoD
Department of ISE
RNSIT

Dr. M K Venkatesha
Principal
RNSIT

Name of the Examiners

External Viva

Signature with Date

1. _____
2. _____

1. _____
2. _____

DECLARATION

I, **Eshwar P V [USN: 1RN18IS043]**, student of VII Semester BE, in Information Science and Engineering, RNS Institute of Technology hereby declare that the Internship work entitled ***BMI Application*** has been carried out by us and submitted in partial fulfillment of the requirements for the *VIII Semester degree of **Bachelor of Engineering in Information Science and Engineering** of Visvesvaraya Technological University, Belagavi* during academic year 2021-2022.

Place : Bengaluru

Date :

**ESHWAR P V
(1RN18IS043)**

ABSTRACT

- The BMI Calculator App is a software application which avoids more manual hours that need to spend in a personally calculate and find Body Mass Index for a particular person at a single click.
- In this work the main scope is to maintain the health.
- The system is developed on android platform using flutter and android studio.
- The BMI app gives information regarding our health when we enter the height, weight and age and we get the information whether we are underweight , overweight or perfect.

ACKNOWLEDGMENT

At the very onset I would like to place our gratefulness to all those people who helped me in making the Internship a successful one.

Coming up, this internship to be a success was not easy. Apart from the sheer effort, the enlightenment of the very experienced teachers also plays a paramount role because it is they who guided me in the right direction.

First of all, I would like to thank the **Management of RNS Institute of Technology** for providing such a healthy environment for the successful completion of internship work.

In this regard, I express sincere gratitude to our beloved Principal **Dr. M K Venkatesha**, for providing us all the facilities.

We are extremely grateful to our own and beloved Professor and Head of Department of Information science and Engineering, **Dr. Suresh L**, for having accepted to patronize me in the right direction with all her wisdom.

We place our heartfelt thanks to **Mrs. Akshata S Bhayyar**, Assistant Professor, Department of Information Science and Engineering for having guided internship and all the staff members of the department of Information Science and Engineering for helping at all times.

I thank **Mr. Akshay D R, Co-Founder & CEO, Enmaz Engineering Services Pvt.Ltd** for providing the opportunity to be a part of the Internship program and having guided me to complete the same successfully.

I also thank our internship coordinator **Dr. R Rajkumar**, Associate Professor, Department of Information Science and Engineering. I would thank my friends for having supported me with all their strength and might. Last but not the least, I thank my parents for supporting and encouraging me throughout. I have made an honest effort in this assignment.

Eshwar P V

TABLE OF CONTENTS

Abstract	i
Acknowledgment	ii
Contents	iii
List of figures	v
List of Abbreviations	vi
1. Introduction	1
1.1 Background	1
1.2 Existing System	1
1.3 Proposed System	1
2. System Design	2
2.1 Widget Tree	2
2.1.1 Questions Screen	2
3. Implementation	3
3.1 Requirement specification	3
3.1.1 Hardware Requirements	3
3.1.2 Software Requirements	3
3.1.3 Flutter	3
3.1.3.1 Dart Platform	3
3.1.3.2 Flutter Engine	4
3.1.3.3 Foundation Library	4
3.1.3.4 Design Specific Widgets	4
3.2 Discussion of Code Segments	4
3.2.1 main.dart	4

4. Testing	
4.1 Introduction	9
4.2 Levels of Testing	9
4.2.1 Unit testing	
4.2.2 Integration Testing	10
4.2.3 System Testing	
4.2.4 Validation Testing	
4.2.5 Output Testing	
4.2.6 User Acceptance Testing	
5. Results	
5.1 a & b	11
5.2 a & b	12
5.3	13
6. Conclusion and future work	13
6.1 Conclusion	13
6.2 Future work	13
References	14

1. Introduction

1.1 Background

Quiz application is an application developed for learning and improving knowledge in the educational area. This application can be used in schools, colleges and also by students themselves before a test or examination to evaluate themselves or even for a quick revision.

1.2 Existing System

In the existing system, the students have to manually contact the teacher to conduct examinations and also, a single examination for a single student cannot be done. It will be tedious for the teacher to when a large number of students want to check their scores. The human effort is more here.

1.3 Proposed System

To overcome the drawbacks mentioned above, the proposed system has been developed. This project has been developed so that a student can easily test themselves and have a quick revision. The interface is easy to understand and is extremely user friendly.

2. System Design

2.1 Widget Tree

2.1.1 Questions Screen

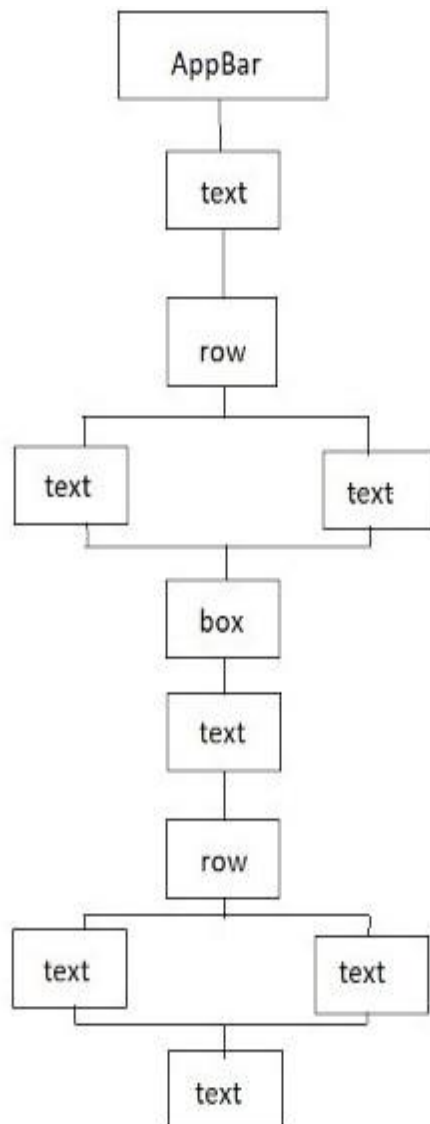


Fig 2.1.1 Widget tree for Questions Screen

1. Implementation

3.1 Requirement Specifications

3.1.1 Hardware Requirements

- CPU: Pentium processor and above
- RAM: 4 GB
- HDD: 40 GB

3.1.2 Software Requirements

- **Operating System:** Windows 8 and above
- **Front-end Design:** Visual Studio Code
- **Front-end Language:** Dart

3.1.3 Flutter

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

The major components of Flutter include:

- Dart Platform
- Flutter Engine
- Foundation Library
- Design-specific widgets
- Flutter Development Tools (DevTools)

3.1.3.1 Dart Platform

Flutter apps are written in the Dart language and make use of many of the language's more advanced features.

On Windows, macOS, and Linux Flutter runs in the Dart virtual machine, which features a just-in-time execution engine. While writing and debugging an app, Flutter uses Just In Time compilation, allowing for "hot reload", with which modifications to source files can be injected into a running application. Flutter extends this with support for stateful, hot

reload, where in most cases changes to source code are reflected immediately in the running app without requiring a restart or any loss of state.

3.1.3.2 Flutter Engine

Flutter's engine, written primarily in C++, provides low-level rendering support using Google's Skia graphics library. Additionally, it interfaces with platform Specific SDKs such as those provided by Android and iOS. The Flutter Engine is a portable runtime for hosting Flutter applications. It implements Flutter's core libraries, including animation and graphics, file and network I/O, accessibility support, plug-in architecture, and a Dart runtime and compile toolchain.

3.1.3.3 Foundation Library

The Foundation library, written in Dart, provides basic classes and functions that are used to construct applications using Flutter, such as APIs to communicate with the engine.

3.1.3.4 Design Specific Widgets

The Flutter framework contains two sets of widgets that conform to specific design languages: Material Design widgets implement Google's design language of the same name, and *Cupertino* widgets implement Apple's iOS Human Interface Guidelines.

3.2 Discussion of Code Segments

3.2.1 main.dart

```
import 'package:flutter/material.dart';
/*
  Green : #12a644
  Grey  : #403f3d
*/
void main() => runApp(
  MaterialApp(
    theme: ThemeData(
      primaryColor:Color(0xFF12a644)
    ),
    home: MyApp(),
    debugShowCheckedModeBanner: false,
  )
);

class MyApp extends StatefulWidget {
  @override
```

```

_MyAppState createState() => _MyAppState();
}

class _MyAppState extends State<MyApp> {
  double _height=170.0;
  double _weight=75.0;
  int _bmi=0;
  String _condition='Select Data';
  @override
  Widget build(BuildContext context) {
    Size size=MediaQuery.of(context).size;
    return Scaffold(
      body: SingleChildScrollView(
        child: Column(
          children: <Widget>[
            Container(
              height: size.height*0.40,
              width: double.infinity,
              decoration: new BoxDecoration(color: Color(0xFF12a644)),
              padding: EdgeInsets.symmetric(vertical: 30.0,horizontal: 10.0),
              child: Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: <Widget>[
                  Text("BMI",style: TextStyle(color: Colors.white,fontWeight:
FontWeight.bold,fontSize: 60.0)),
                  Text("Calculator",style: TextStyle(color: Colors.white,fontSize: 40.0)),
                  SizedBox(
                    width: double.infinity,
                    child: Container(
                      child:Text("$_bmi",
                        style:TextStyle(
                          color: Colors.white,
                          fontWeight: FontWeight.bold,
                          fontSize: 45.0
                        ),textAlign: TextAlign.right,
                      ),
                    ),
                  ),
                  RichText(
                    text: TextSpan(
                      text: "Condition : ",
                      style: TextStyle(
                        color: Colors.white,
                        fontSize: 25.0
                      ),
                    ),
                    children: <TextSpan>[
                      TextSpan(
                        text: "$_condition",
                        style: TextStyle(
                          color: Colors.white,
                          fontSize: 25.0,
                          fontWeight: FontWeight.bold
                        ),
                      ),
                    ],
                  ),
                ],
              ),
            ),
          ],
        ),
      ),
    );
  }
}

```

```

    )
  ],
),
),
Container(
  padding: EdgeInsets.symmetric(horizontal: 10.0,vertical: 10.0),
  width: double.infinity,
  child: Column(
    children: <Widget>[
      SizedBox(height: size.height*0.03,),
      Text("Choose Data",style: TextStyle(color: Color(0xFF12a644),fontSize:
30.0,fontWeight: FontWeight.bold),),
      SizedBox(height: size.height*0.03,),
      RichText(
        text: TextSpan(
          text: "Height : ",
          style: TextStyle(
            color: Color(0xFF403f3d),
            fontSize: 25.0
          ),
        ),
        children: <TextSpan>[
          TextSpan(
            text: "$_height cm",
            style: TextStyle(
              color: Color(0xFF403f3d),
              fontSize: 25.0,
              fontWeight: FontWeight.bold
            ),
          ),
        ],
      ),
    ],
  ),
  SizedBox(height: size.height*0.03,),
  Slider(
    value: _height,
    min:0,
    max: 250,
    onChanged: (height){
      setState() {
        _height=height;
      };
    },
    divisions: 250,
    label: "$_height",
    activeColor:Color(0xFF403f3d),
    inactiveColor: Colors.grey,
  ),
  SizedBox(height: size.height*0.03,),
  RichText(
    text: TextSpan(
      text: "Weight : ",
      style: TextStyle(
        color: Color(0xFF403f3d),
        fontSize: 25.0
      ),
    ),
  ),

```

```

        children: <TextSpan>[
          TextSpan(
            text: "$_weight kg",
            style: TextStyle(
              color: Color(0xFF403f3d),
              fontSize: 25.0,
              fontWeight: FontWeight.bold
            ),)
        ]
      ),
    ),
    SizedBox(height: size.height*0.03,),
    Slider(
      value: _weight,
      min:0,
      max: 300,
      onChanged: (weight){
        setState() {
          _weight=weight;
        };
      },
      divisions: 300,
      label: "$_weight",
      activeColor:Color(0xFF403f3d),
      inactiveColor: Colors.grey,
    ),
    SizedBox(height: size.height*0.03,),
    Container(
      width: size.width*0.8,
      child: ClipRRect(
        borderRadius: BorderRadius.circular(30.0),
        child: FlatButton(
          onPressed: (){
            setState() {
              //18.5 - 25 Healthy 25-30 Overweight >30 Obesity
              _bmi=( _weight/(( _height/100)*( _height/100))).round().toInt();
              if(_bmi>=18.5 && _bmi<=25) {_condition=" Normal";}
              else if(_bmi>25 && _bmi<=30) {_condition=" Overweight";}
              else if(_bmi>30) {_condition=" Obesity";}
              else {_condition=" Underweight";}
            };
          },
          child: Text("Calculate",style: TextStyle(color: Colors.white,fontSize: 20.0)),
          color:Color(0xFF12a644),
          padding: EdgeInsets.symmetric(vertical: 15,horizontal: 40),
        ),
      ),
    ),
  ],
),
),
); }

```

4. Testing

2.1 Introduction

Testing is a process of executing a program with the interest of finding an error. A good test is one that has high probability of finding the yet undiscovered error. Testing should systematically uncover different classes of errors in a minimum amount of time with a minimum number of efforts. Two classes of inputs are provided to test the process

- A software configuration that includes a software requirement specification, a design specification and source code.
- A software configuration that includes a test plan and procedure, any testing tool and test cases and their expected results.

2.2 Levels of Testing

2.2.1 Unit Testing

Unit testing is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output.

Unit testing is commonly automated, but may still be performed manually. The objective in unit testing is to isolate a unit and validate its correctness. A manual approach to unit testing may employ a step-by-step instructional document. The unit testing is the process of testing the part of the program to verify whether the program is working correct or not. In this part the main intention is to check the each and every input which we are inserting to our file. Here the validation concepts are used to check whether the program is taking the inputs in the correct format or not.

Unit testing may reduce uncertainty in the units themselves and can be used in a bottom-up testing style approach. By testing the parts of a program first and then testing the sum of its parts, integration testing becomes much easier. Unit test cases embody characteristics that are critical to the success of the unit.

2.2.2 Integration Testing

Integration testing is also taken as integration and testing this is the major testing process where the units are combined and tested. Its main objective is to verify whether the major parts of the program is working fine or not. This testing can be done by choosing the options in the program and by giving suitable inputs.

2.2.3 System Testing

System testing is defined as testing of a complete and fully integrated software product. This testing falls in black-box testing wherein knowledge of the inner design of the code is not a pre-requisite and is done by the testing team. System testing is done after integration testing is complete. System testing should test functional and non-functional requirements of the software.

2.2.4 Validation Testing

In this, requirements established as part of software requirements analysis are validated against the software that has been constructed. Validation testing provides final assurance that software meets all functional, behavioral and performance requirements. Validation can be defined in many ways but a simple definition is that validation succeeds when software Function in a manner that can be reasonably by the customer.

1. Validation test criteria
2. Configuration review
3. Alpha and Beta testing (conducted by end user)

2.2.5 Output Testing

After preparing test data, the system under study is tested using the test data. While testing the system using test data, errors are again uncovered and corrected by using above testing and corrections are also noted for future use.

2.2.6 User Acceptance Testing

User acceptance testing is a type of testing performed by the end user or the client to verify/accept the software application to the production environment.

UAT is done in the final phase of testing.

5. Results

5.1 Start Screen

10:04 5.82 49%

BMI Calculator

0

Condition : **Select Data**

Choose Data

Height : **170.0 cm**

Weight : **75.0 kg**

Calculate

5.2 To Display Normal BMI

10:10 24% 50%

BMI Calculator

23

Condition : **Normal**

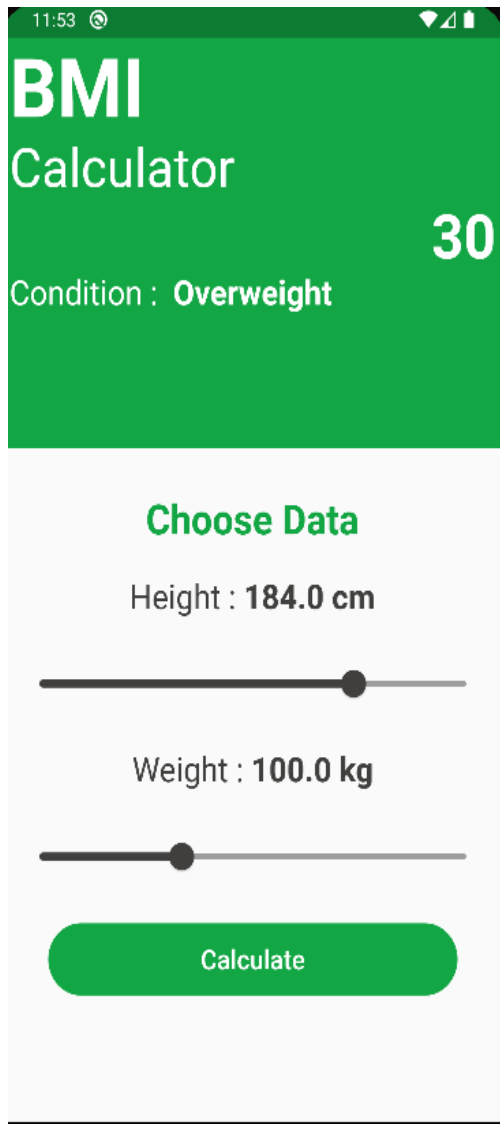
Choose Data

Height : **163.0 cm**

Weight : **62.0 kg**

Calculate

5.3 To Display Overweight BMI



11:53

BMI Calculator

30

Condition : **Overweight**

Choose Data

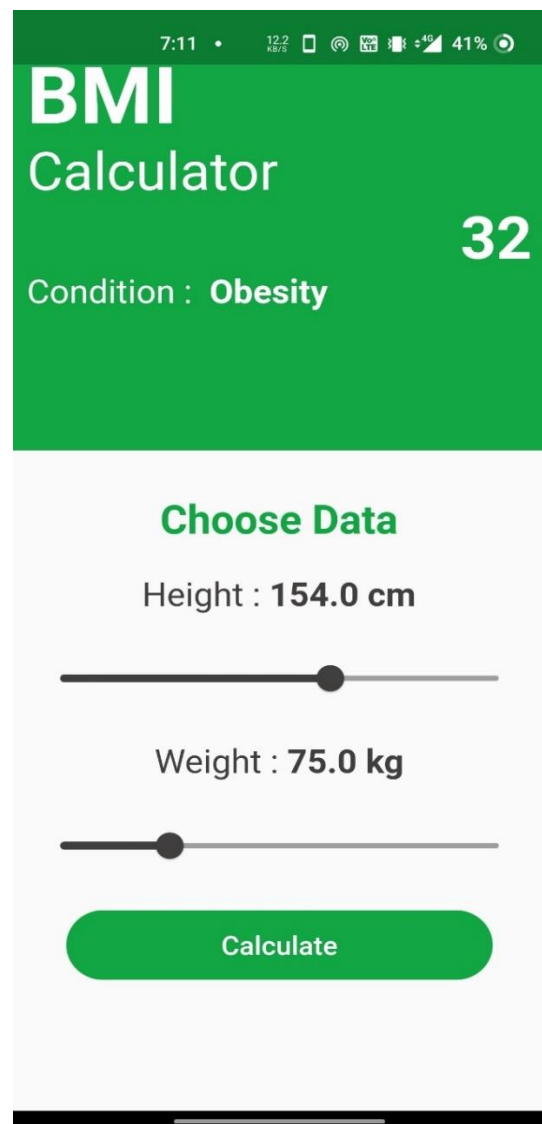
Height : 184.0 cm

Weight : 100.0 kg

Calculate

This screenshot shows the BMI Calculator app interface. The top status bar displays the time 11:53. The app title "BMI Calculator" is at the top. The calculated BMI value "30" is shown in large text. Below it, the condition "Overweight" is displayed. The input section, titled "Choose Data", shows a height of 184.0 cm and a weight of 100.0 kg, each with a slider control. A green "Calculate" button is at the bottom.

5.4 To Display Obesity BMI



7:11 • 12.2 KB/s

BMI Calculator

32

Condition : **Obesity**

Choose Data

Height : 154.0 cm

Weight : 75.0 kg

Calculate

This screenshot shows the BMI Calculator app interface. The top status bar displays the time 7:11 and network speed 12.2 KB/s. The app title "BMI Calculator" is at the top. The calculated BMI value "32" is shown in large text. Below it, the condition "Obesity" is displayed. The input section, titled "Choose Data", shows a height of 154.0 cm and a weight of 75.0 kg, each with a slider control. A green "Calculate" button is at the bottom.

5.5 To Display Underweight BMI

A mobile application interface for a BMI calculator. The top section has a green background with the text "BMI Calculator" in white. Below this, the condition is displayed as "Condition : Underweight" in white. A large white number "17" is shown in the top right corner. The bottom section has a light gray background with the heading "Choose Data" in green. It contains two sliders: "Height : 154.0 cm" and "Weight : 40.0 kg". Below the sliders is a green button labeled "Calculate".

5.6 Web Version

A web browser interface for a BMI calculator. The browser address bar shows "localhost:50509/#/". The page has a green header with "BMI Calculator" in white. Below the header, the condition is "Condition : Select Data" in white. A large white number "0" is in the top right corner. The main content area has a light gray background with the heading "Choose Data" in green. It features two sliders: "Height : 170 cm" and "Weight : 75 kg". At the bottom is a green button labeled "Calculate".

6.Conclusion and Future Work

6.1 Conclusion

- The package was designed in such a way that future modifications can be done easily.
- The following conclusion can be deduced from development of project.
- Automation of the entire system improves the efficiency
- It gives appropriate access to the authorized users depending on their permission
- It effectively overcomes the time complexity.

6.2 Future Enhancement

- This app avoids the manual work and the problems concern with it. Centralized management of the database and one app to manage the BMI Calculator of different section of people.
- We can add alter message to the person to eat and exercise according to their BMI.

References

- [1] <https://flutter.dev/>
- [2] <https://developers.google.com/learn/pathways/intro-to-flutter>
- [3] <https://www.youtube.com/>
- [4] Beginning Flutter: A Hands On Guide to App Development by Marco L Napoli