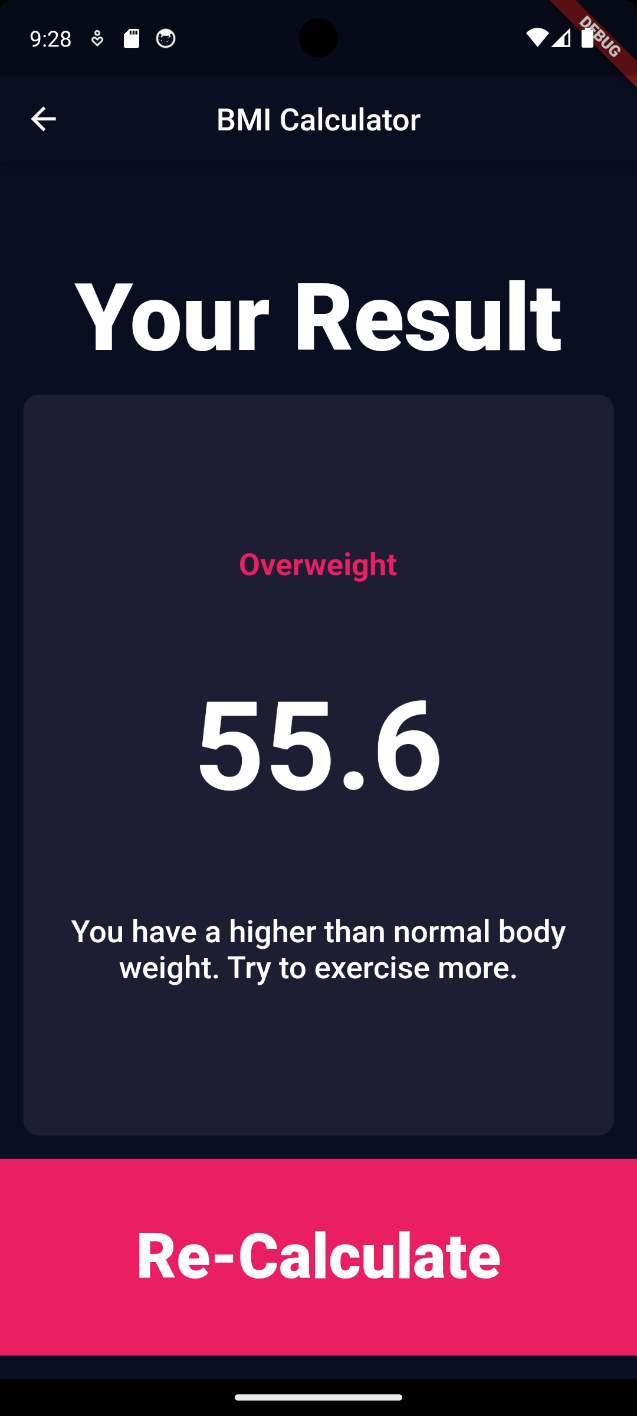
**BMI Calculator**

Flutter Preview:



Code Explanation (step by step):

1. Setup Flutter Projects -> go to main.dart files -> delete all values
2. Set up the main Function

import 'package:flutter/material.dart';

import 'package:my\_app/main\_screen.dart';

void main() {

  runApp(

    MaterialApp(

      theme: ThemeData.dark().copyWith(

        primaryColor: const Color(0xFF0A0E21),

        scaffoldBackgroundColor: const Color(0xFF0A0E21),

      ),

      home: const MainScreen(),

    ),

  );

}

1. ‘theme:’ is a property that has a special function to change the theme of the scaffold page. There we use ThemeData.dark() to change the theme to dark mode, and copywith() to change the settings of some theme properties. Here, we change the primary color and set the background color to a more black-blueish color,
2. Make a new widget file named MainScreen() widget where this is a stateful widget. Because here, the UI will have a rendering purpose and we also want to change to another screen when we calculate. Before, we only change the content by rendering the UI but now, we will change to another screen. More explanation in step(‘’). The goal of the main screen is to make this UI:

A screenshot of a cell phone

Description automatically generated

1. In the MainScreen widget, make the UI off the application, here the UI is a bit tricky since there is a lot of complex row and column usage. First make the App Bar that has a name of BMI Calculator and center it by using ‘centerTitle: true’. Then change the background color so it matches the background of the body.
2. Let’s analyze the UI, it has a column there there is a column section that stores the gender selection, the height section, the weight and age section, and the calculate button section. All of these section (except for the calculate button section) will be in a separate file to make the code readable. We will name the files in this format:

A screenshot of a computer

Description automatically generated

1. What male and female consist of? 2 types that cannot be changed/ a fixed type. Just like in life, there are only 2 types of gender, not more not less. For this scenario, we need to make another class in another file that stores a value with those 2 properties. Use enum for establishing a datatype that has a fixed properties such as this format:

enum Gender {

  male,

  female,

}

When the file is imported, we can use a Gender class to set the variables as a male/ female only for example: Gender.male or Gender.female

1. In the male\_female\_widgets.dart, make a stateful widget that has stores the male/ female button. We will need a row and 2 containers for making the curved borders and the male and female icon and texts. Do not forget to wrap the container with a gesture button so that it can be clicked.
2. Make a function that when a male is tapped, the color changes and when the female button is tapped, the female button changes and the male button change to the original and vice versa. Remember every click determines whether the user’s input is a male/ female. So we will need a variable that has a type of ‘Color’ and set both color the same. In the code, you will see a onTapMaleButton and onTapFemaleButton function.
3. In the function, you will also see a widget.getGender(Gender.male). Now what does that mean? to know the meaning we first must go back to the main\_screen.dart file and make a void function called getGender. If you learn OOP you will recognize that we can use a getter and setter function to get a value from other files/ classes. Here, we must make a variable for the currentGender chosen and a parameter for the function that extracts the value from male\_female widgets. Then, use a contructor in the function so that we can pass the function to another class and run it in other classes too.

void getGender(Gender chosenGender) {

    currentGender = chosenGender;

  }

1. Do the same steps for the height, age, and weight. Then, make a slider\_widget that has a constructor of a void Function getHeight that responsible to get the data from the slider\_widget and pass it to the main\_screen widget. To make a slider, we can use a **Slider** **Widget** that has a property of value, min, max, and on Changes status. Value means we must set an initial value every time the screen is load. In the code, we can make another int variable that keeps the initial value and the value when the slider slides. Value property only accept double datatypes so we must change the datatype to double by using .toDouble(), this is also the same to output the value result to the screen by using .toString() since text only can accept string parameters. Min, max is a limiter for the slider and onChanged will define the function that runs when the slider slides. In this code, we will render the text by using set state and update that value, then pass the value to the getter function.
2. For weight, we will do similar things, just a tricky UI that has a row 2 containers, each container will have a column that defines the title and the value of the weight and lastly another row to define the ‘+’ and ‘-‘ icon.
3. The logic and method is the same as step ‘l’. When the ‘+’ button is pressed, then the value increases and when ‘-‘ is pressed, the value decreases and send the updated value to the getter function.
4. Lastly, the last section was the calculate button which takes the rest of the bottom screen. So, we can use expanded to make as much space available using container and have a text centred. Do not forget to add a Gesture Detector so that when pressed, it will calculate the BMI and show the result in another screen.
5. Make a void function in the main\_screen.dart file that calculates the result and then navigate to other screen. To navigate to other screen, first make a result\_screen.dart file which is a stateless widget that has a scaffold as it’s first parent widget. Then we do not make the UI yet, just a dummy screen that will change later. In the goToResultScreen() (see the picture below), calculate the result, then using a method named ‘**Navigator.push**’, we can change the screen to another screen. The mechanism is that the main\_screen will be stacked by another screen that we wanted to show. Here, we set the context, then the materialPageRoute which has a builder context that returns the screen we want to show. Here we wanted to show the Result\_screen.dart. we have (result:result) here that states the results value in the ResultScreen() should have the same value as in the mainScreen().

double CalculateBMI(currentHeight, currentWeight) {

    return currentWeight / pow(currentHeight / 100, 2);

  }

  void goToResultScreen() {

    double result = CalculateBMI(currentHeight, currentWeight);

    Navigator.push(

      context,

      MaterialPageRoute(

        builder: (context) {

          return ResultScreen(result: result);

        },

      ),

    );

  }

1. In the resultScreen, our objective is to make a screen such as this:

A screenshot of a cell phone

Description automatically generatedThe backbutton on the top left is automatically generated by flutter when we navigate the screen so that we do not need to manually make a function to go back. Thou in this UI, we will make it manually since we cant to re-calculate the BMI that means we must go back to the main screen.

The ’overweight’ shown in the picture is a string that has a function to show whether you are ‘underweight’, ‘normal’ or ‘overweight’. To determine it, we must use an if statement based on the BMI value with a String parameter since we will return a string. You can see the example in the code. This also goes the same way for the ‘you have a higher than normal body…’ that has a comment text shown. The ’55.6’ is the BMI result which just needs to be showed.

To Re-Calculate a.k.a. to go back to the main screen, we can use ‘**Navigator.pop**’ method. Which is just a simple code. Make sure to use it then needed such as put the function on a button/ void.

onTap: () {

                  Navigator.pop(context);

                },

GitHub Link: