**PYTHON PROJECT**

**TITLE: BMI CALCULATOR**

**NAME: ROHIT DEEPAK BHAVSAR**

**ROLL NO : 05**

**DATE: 15-12-2023**

**Abstract**

This report provides an overview of the Python project titled "BMI CALCULATOR" BMI (Body Mass Index) calculator using Python involves creating a function or a class that takes in the necessary inputs (weight and height) and returns the calculated BMI,outlining its objectives, methodology, and key findings.

**Table of Contents**

1. Introduction
2. Project Overview
3. Methodology
4. Implementation code
5. Results screen shots and Report/Analysis
6. Conclusion
7. References

**INTRODUCTION**

BMI is a measure of body fat based on an individual's weight and height. It is commonly used to categorize individuals into different weight status categories, such as underweight, normal weight, overweight, and obesity. It provides a simple numerical representation that helps categorize people into different weight status groups

BMI values are typically interpreted as follows:

**Underweight:** BMI less than 18.5

**Normal weight:** BMI between 18.5 and 24.9

**Overweight:** BMI between 25 and 29.9

**Obese:** BMI of 30 or greater

**PROJECT OVERVIEW**

BMI Calculator project aims to create a simple yet effective tool that allows users to calculate their BMI based on their weight and height. The program will not only compute the BMI but also interpret the result, providing information about the user's weight status category (underweight, normal weight, overweight, or obese).

The project will be implemented in Python, utilizing functions for modularity and clarity. The main functionalities will include:

calculate\_bmi(weight, height):

A function that takes weight and height as input and returns the calculated BMI using the BMI formula.

interpret\_bmi(bmi):

A function that takes the calculated BMI as input and categorizes it into different weight status categories.

main():The main function that orchestrates the flow of the program, taking user input, calling the calculation and interpretation functions, and displaying the result.

**METHODOLOGY**

The development of a BMI (Body Mass Index) calculator involves a systematic methodology to ensure the creation of an accurate, reliable, and user-friendly tool.

* **Objective:**
  + Developing a BMI calculator for accurate and user-friendly body mass assessment.
* **Considerations:**
  + Defining scope and target audience.
  + Choosing the platform and programming language (e.g., Python).
* **Tasks:**
  + Understanding BMI formula and interpretation.
  + Familiarize with BMI categories and health implications.
* **Features:**
  + Name and gender input
  + Weight and height input.
  + BMI calculation.
  + Result interpretation.
  + Additional features (input validation, health recommendations).
* **Coding Tasks:**
  + Write BMI calculation functions.
  + Implement result interpretation.
  + Develop the main function for user interaction.
  + Include input validation mechanisms.
* **Testing Stages:**
  + Unit testing for individual functions.
  + Integration testing for function interactions.
  + Scenario testing for various input scenarios.
  + Address edge cases.
* **Content:**
  + Explanation of BMI formula.
  + User instructions.
  + Details on weight status categories.
  + Additional features and recommendations.
* **Monitoring Activities:**
  + Monitor user feedback and bug reports.
  + Address issues promptly through updates.
* **Considerations for Future Versions:**
  + Graphical representations.
  + Historical tracking.
  + Integration with other health metrics.

By following this systematic methodology, the development of the BMI calculator can proceed in an organized and efficient manner, ensuring a robust and valuable tool for users.

Top of Form

By incorporating these considerations into the methodology, the development process becomes comprehensive, addressing not only the technical aspects but also usability, security, and legal compliance. This holistic approach ensures the creation of a BMI calculator that is not only accurate but also reliable, secure, and user-friendly.

**IMPLEMENTATION OF CODE**

import tkinter

import sqlite3

from tkinter import \*

from tkinter import messagebox

def save\_data(gender, height, weight, bmi):

conn = sqlite3.connect('data1.db')

c = conn.cursor()

c.execute('''CREATE TABLE IF NOT EXISTS bmi\_data

(gender varchar(10),

height REAL,

weight REAL,

bmi REAL)''')

c.execute("INSERT INTO bmi\_data (gender, height, weight, bmi) VALUES (?, ?, ?, ?, ?)",

(gender, height, weight, bmi))

conn.commit()

conn.close()

def reset\_entry():

age.delete(0, 'end')

height.delete(0, 'end')

weight.delete(0, 'end')

def calculate\_bmi():

kg = float(weight.get())

m = float(height.get()) / 100

bmi = kg / (m \* m)

bmi = round(bmi, 1)

bmi\_index(bmi)

def bmi\_index(bmi):

if bmi < 18.5:

messagebox.showinfo('Results', f'BMI = {bmi} is Underweight')

elif 18.5 <= bmi < 24.9:

messagebox.showinfo('Results', f'BMI = {bmi} is Normal')

elif 24.9 <= bmi < 29.9:

messagebox.showinfo('Results', f'BMI = {bmi} is Overweight', icon='warning')

elif bmi >= 29.9:

messagebox.showinfo('Results', f'BMI = {bmi} is Obesity', icon='warning')

else:

messagebox.showerror('Results', 'something went wrong!')

ws = Tk()

ws.title('Results')

ws.geometry('300x350')

ws.config(bg='lightblue')

var = IntVar()

frame = Frame(

ws,

padx=25,

pady=25

)

frame.pack(expand=True)

age\_lb = Label(

frame,

text="Enter Age (2 - 120)", bg='yellow'

)

age\_lb.grid(row=1, column=1)

age = Entry(

frame,

)

age.grid(row=1, column=2, pady=5)

gen\_lb = Label(

frame,

text='Select Gender', bg='yellow'

)

gen\_lb.grid(row=2, column=1)

frame2 = Frame(

frame

)

frame2.grid(row=2, column=2, pady=5)

male\_rb = Radiobutton(

frame2,

text='Male',

variable=var,

value=1

)

male\_rb.pack(side=LEFT)

female\_rb = Radiobutton(

frame2,

text='Female',

variable=var,

value=2

)

female\_rb.pack(side=RIGHT)

height\_lb = Label(

frame,

text="Enter Height (cm) ", bg='yellow'

)

height\_lb.grid(row=3, column=1)

weight\_lb = Label(

frame,

text="Enter Weight (kg) ", bg='yellow'

)

weight\_lb.grid(row=4, column=1)

height = Entry(

frame,

)

height.grid(row=3, column=2, pady=5)

weight = Entry(

frame,

)

weight.grid(row=4, column=2, pady=5)

frame3 = Frame(

frame

)

frame3.grid(row=5, columnspan=3, pady=10)

cal\_btn = Button(

frame3,

text='Calculate', bg='pink',

command=calculate\_bmi

)

cal\_btn.pack(side=LEFT)

reset\_btn = Button(

frame3,

text='Reset', bg='pink',

command=reset\_entry

)

reset\_btn.pack(side=LEFT)

exit\_btn = Button(

frame3,

text='Exit', bg='pink',

command=lambda: ws.destroy()

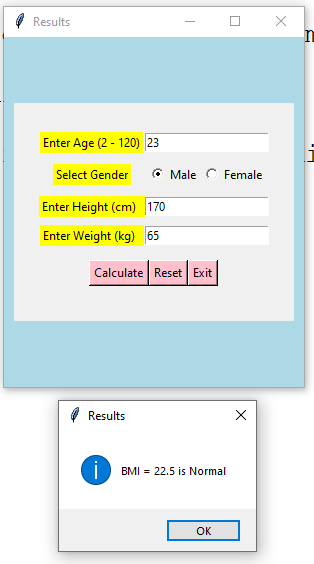
)

exit\_btn.pack(side=RIGHT)

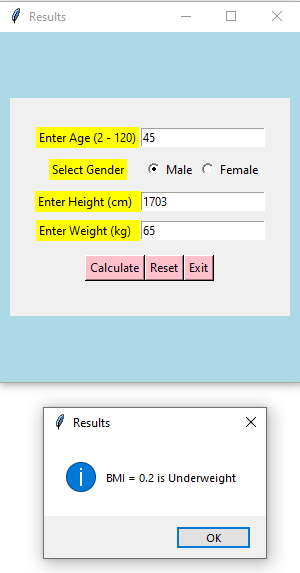
ws.mainloop()

**RESULTS OF CODE**

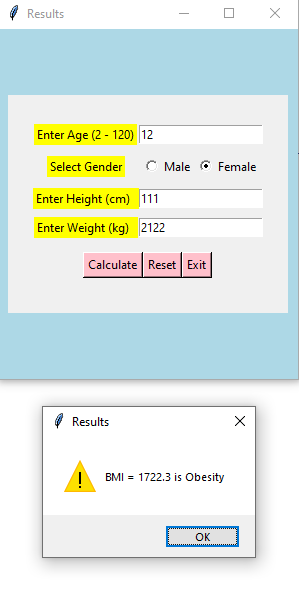
* + 1. **When the result come normal.**



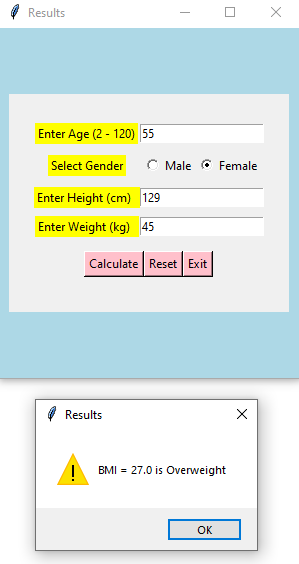
* + 1. **When the result is underweight**

****

* + 1. **When the result is obese.**



* + 1. **When the result is overweight**



**CONCLUSION**

In conclusion, developing a BMI (Body Mass Index) calculator involves a thoughtful and systematic process to ensure it serves its purpose effectively. Starting with planning and research, we define the goals and understand the BMI concept. The design phase helps us create an interface that is easy to use and understand.

Throughout the implementation, we focus on accuracy in BMI calculation, taking care to handle user inputs properly. Usability testing and accessibility considerations ensure that the calculator is user-friendly and accessible to a diverse audience.

Normally, this calculator helps to know about the fitness of health. Like whether the person is healthy, underweight, overweight like that. This helps a lot and because of this calculator we can change our eating habits to come in the bracket of normal or perfect section.

**REFERENCES**

<https://www.geeksforgeeks.org/>

<https://www.w3schools.com/>

<https://www.codecademy.com/>

<https://www.coursera.org/>

<https://www.codementor.io/>

<https://stackoverflow.com/>

<https://www.sololearn.com/en/>