BMI- BODY WEIGHT INDEX

```
import customtkinter as ctk
import tkinter as tk
from settings import *


class App(ctk.CTk):
    def __init__(self):

        # window setup
        super().__init__(fg_color=GREEN)
        self.title('')
        row_img = tk.PhotoImage(file='/home/saabir/Music/Tkinter/GUI-development-tkinter/QR-Code Project/empty.png')
        self.tk.call('wm', 'iconphoto', self._w, row_img)
        self.geometry('400x400')
        self.resizable(False, False)

        # layout
        self.columnconfigure(0, weight=1)
        self.rowconfigure((0, 1, 2, 3), weight=1, uniform='a')

        # data
        self.height_int = ctk.IntVar(value=170)
        self.weight_float = ctk.DoubleVar(value=65)
        self.bmi_string = ctk.StringVar()
        self.update_bmi()

        # tracing
        self.height_int.trace('w', self.update_bmi)
        self.weight_float.trace('w', self.update_bmi)

        # widgets
        ResultText(self, self.bmi_string)
        self.weight_input = WeightInput(self, self.weight_float)
        self.height_input = HeightInput(self, self.height_int)

        self.mainloop()

    def change_units(self, *args):
        self.height_input.update_text(self.height_int.get())
        self.weight_input.update_weight()

    def update_bmi(self, *args):
        height_meter = self.height_int.get() / 100
        weight_kg = self.weight_float.get()
        bmi_result = round(weight_kg / height_meter ** 2, 2)
        self.bmi_string.set(bmi_result)



class ResultText(ctk.CTkLabel):
    def __init__(self, parent, bmi_string):
        font = ctk.CTkFont(family=FONT, size=MAIN_TEXT_SIZE, weight='bold')
        super().__init__(master=parent, text=22.5, font=font, text_color=WHITE, textvariable=bmi_string)
        self.grid(column=0, row=0, rowspan=2, sticky='nsew')
```

```
class WeightInput(ctk.CTkFrame):
    def __init__(self, parent, weight_float):
        super().__init__(master=parent, fg_color=WHITE)
        self.grid(column=0, row=2, sticky='nsew', padx=10, pady=10)
        self.weight_float = weight_float

        # output logic
        self.output_string = ctk.StringVar()
        self.update_weight()

        # layout
        self.rowconfigure(0, weight=1, uniform='b')
        self.columnconfigure(0, weight=2, uniform='b')
        self.columnconfigure(1, weight=1, uniform='b')
        self.columnconfigure(2, weight=3, uniform='b')
        self.columnconfigure(3, weight=1, uniform='b')
        self.columnconfigure(4, weight=2, uniform='b')

        # text
        font = ctk.CTkFont(family=FONT, size=INPUT_FONT_SIZE)
        label = ctk.CTkLabel(self, textvariable=self.output_string, text_color=BLACK, font=font)
        label.grid(row=0, column=2)

        # buttons
        minus_button = ctk.CTkButton(self, command=lambda: self.update_weight(('minus', 'large')), text='-', font=font,
                                     text_color=BLACK, fg_color=LIGHT_GRAY, hover_color=GRAY,
                                     corner_radius=BUTTON_CORNER_RADIUS)
        minus_button.grid(row=0, column=0, sticky='ns', padx=8, pady=8)

        plus_button = ctk.CTkButton(self, command=lambda: self.update_weight(('plus', 'large')), text='+', font=font,
                                    text_color=BLACK, fg_color=LIGHT_GRAY, hover_color=GRAY,
                                    corner_radius=BUTTON_CORNER_RADIUS)
        plus_button.grid(row=0, column=4, sticky='ns', padx=8, pady=8)

        small_plus_button = ctk.CTkButton(self, command=lambda: self.update_weight(('plus', 'small')), text='+',
                                          font=font, text_color=BLACK, fg_color=LIGHT_GRAY, hover_color=GRAY,
                                          corner_radius=BUTTON_CORNER_RADIUS)
        small_plus_button.grid(row=0, column=3, padx=4, pady=4)

        small_minus_button = ctk.CTkButton(self, command=lambda: self.update_weight(('minus', 'small')), text='-',
                                           font=font, text_color=BLACK, fg_color=LIGHT_GRAY, hover_color=GRAY,
                                           corner_radius=BUTTON_CORNER_RADIUS)
        small_minus_button.grid(row=0, column=1, padx=4, pady=4)

    def update_weight(self, info=None):
        if info:
            amount = 1 if info[1] == 'large' else 0.1
            if info[0] == 'plus':
                self.weight_float.set(self.weight_float.get() + amount)
            else:
                self.weight_float.set(self.weight_float.get() - amount)

        self.output_string.set(f'{round(self.weight_float.get(), 1)}kg')
```

```
class HeightInput(ctk.CTkFrame):
    def __init__(self, parent, height_int):
        super().__init__(master=parent, fg_color=WHITE)
        self.grid(row=3, column=0, sticky='nsew', padx=10, pady=10)

        # widgets
        slider = ctk.CTkSlider(
            master=self,
            command=self.update_text,
            button_color=GREEN,
            button_hover_color=GRAY,
            progress_color=GREEN,
            fg_color=LIGHT_GRAY,
            variable=height_int,
            from_=100,
            to=250)
        slider.pack(side='left', fill='x', expand=True, pady=10, padx=10)

        self.output_string = ctk.StringVar()
        self.update_text(height_int.get())

        output_text = ctk.CTkLabel(self, textvariable=self.output_string, text_color=BLACK,
                                   font=ctk.CTkFont(family=FONT, size=INPUT_FONT_SIZE))
        output_text.pack(side='left', padx=20)

    def update_text(self, amount):
        height = amount / 100
        self.output_string.set(f'{height:.2f}m')


if __name__ == '__main__':
    App()


# SETTINGS
# text sizes
FONT = 'Calibri'
MAIN_TEXT_SIZE = 150
INPUT_FONT_SIZE = 26
SWITCH_FONT_SIZE = 18
BUTTON_CORNER_RADIUS = 6

# colors
GREEN = '#50BFAB'
DARK_GREEN = '#3A8A7B'
WHITE = '#F2F2F2'
BLACK = '#1F1F1F'
LIGHT_GRAY = '#E8E8E8'
GRAY = '#D9D9D9'

# title hex
TITLE_HEX_COLOR = 0x00ABBF50 # hex order: 0x00bbggrr #50BFAB
```