Project Title: Pediatric Dose Calculator

SAT 4650 FINAL PROJECT

BY IFUNANYA EZEUMEH

Project description

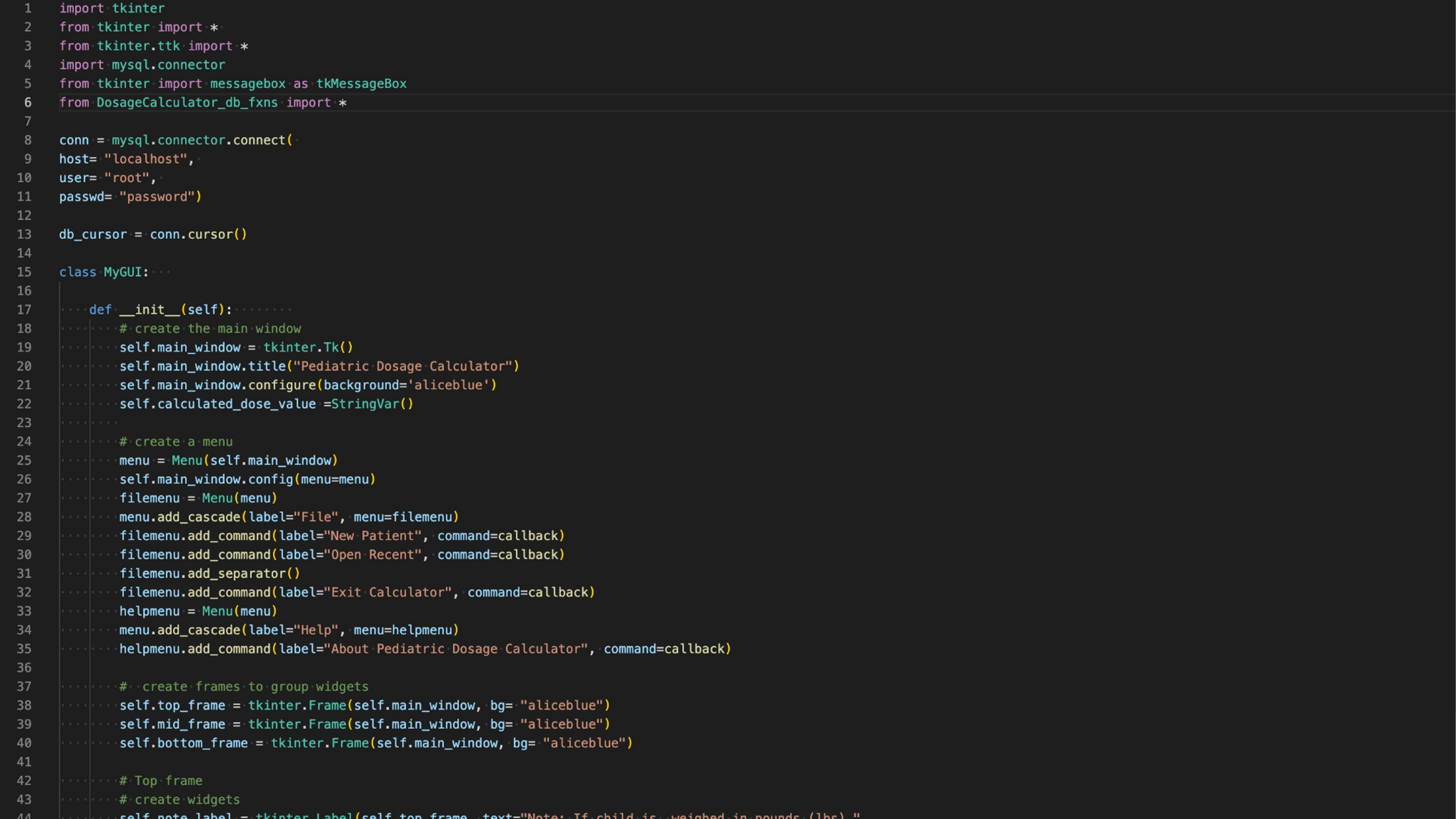
- Plan: To develop a Python program that will help calculate compounding doses for pediatric populations.
- Modules: Functions, Database & GUI.
- Dataflow/ major functionalities:
  - Users will use entry widget to input patient relevant information.
  - Formulas to be used to calculate the doses will be defined with functions and stored to a database.
  - The calculated doses will be outputted in an info dialog box.
- Input: Relevant patient information, and medication name
- Output: Calculated compounding dose

Major Application of The Pediatric Dosage Calculator

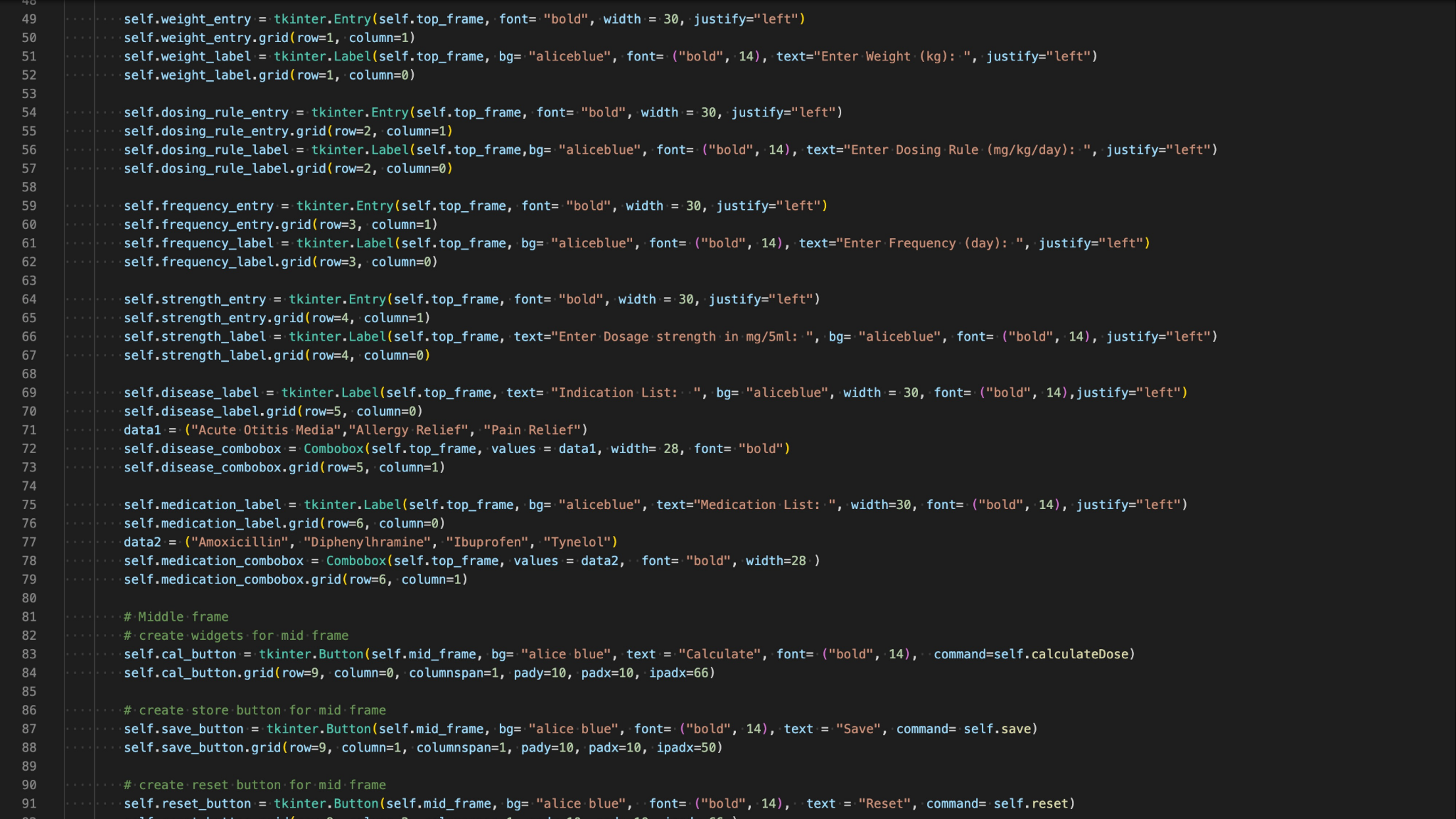
Children of the same age grow at different rates. For children in the same age group, there are wide variations in their weight . Consequently, dosages are usually calculated using their weight. It is important to ensure that every child is getting the optimum and safe dose required for them to receive quality healthcare.

Long Term Goals

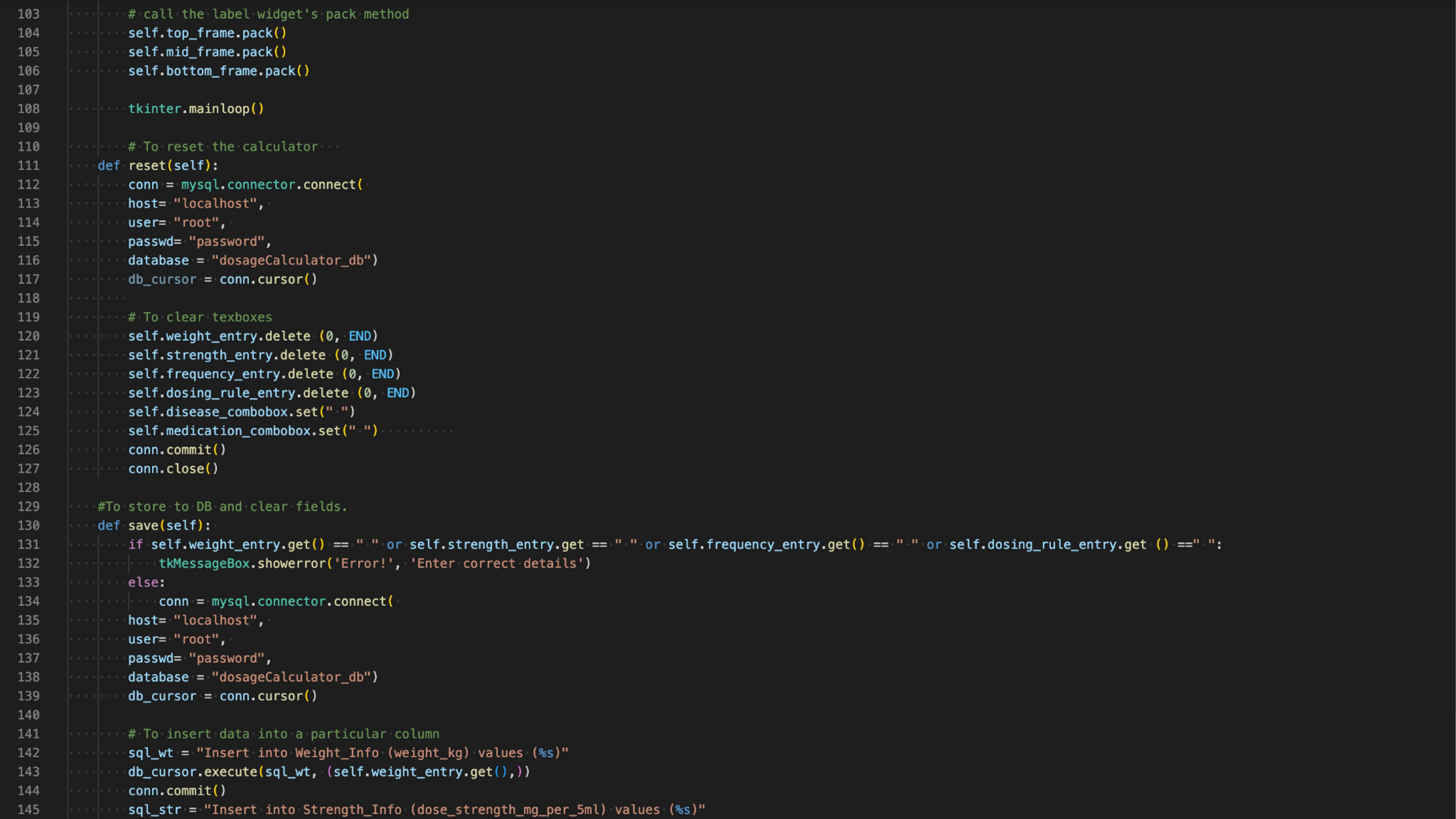
To expand the functions of the calculator to be able to calculate doses for other special populations. Thereafter, create an Application Programming Interface (API) that can be embedded into a Pharmacy Health Information Technology (HIT) solution which will help Pharmacists quickly determine doses required for many special populations.



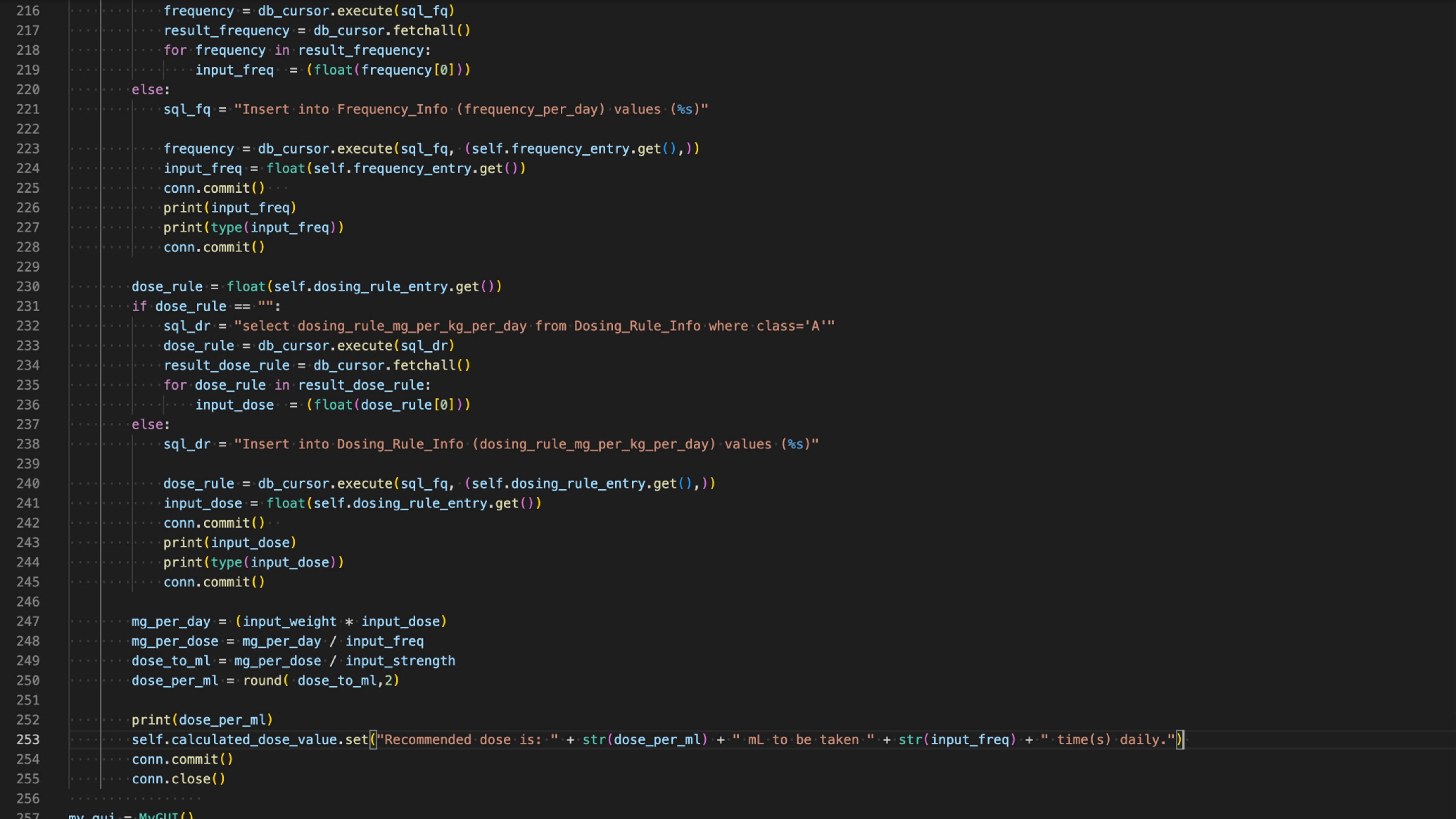
```
import tkinter
from tkinter import *
from tkinter.ttk import *
import mysql.connector
from tkinter import messagebox as tkMessageBox
from DosageCalculator_db_fxns import *

conn = mysql.connector.connect(
host= "localhost",
user= "root",
passwd= "password")

db_cursor = conn.cursor()

class MyGUI:

    def __init__(self):
        # create the main window
        self.main_window = tkinter.Tk()
        self.main_window.title("Pediatric Dosage Calculator")
        self.main_window.configure(background='aliceblue')
        self.calculated_dose_value =StringVar()

        # create a menu
        menu = Menu(self.main_window)
        self.main_window.config(menu=menu)
        filemenu = Menu(menu)
        menu.add_cascade(label="File", menu=filemenu)
        filemenu.add_command(label="New Patient", command=callback)
        filemenu.add_command(label="Open Recent", command=callback)
        filemenu.add_separator()
        filemenu.add_command(label="Exit Calculator", command=callback)
        helpmenu = Menu(menu)
        menu.add_cascade(label="Help", menu=helpmenu)
        helpmenu.add_command(label="About Pediatric Dosage Calculator", command=callback)

        #  create frames to group widgets
        self.top_frame = tkinter.Frame(self.main_window, bg= "aliceblue")
        self.mid_frame = tkinter.Frame(self.main_window, bg= "aliceblue")
        self.bottom_frame = tkinter.Frame(self.main_window, bg= "aliceblue")

        # Top frame
        # create widgets
        self.note_label = tkinter.Label(self.top_frame, text="Note: If child is  weighed in pounds (lbs),"
```



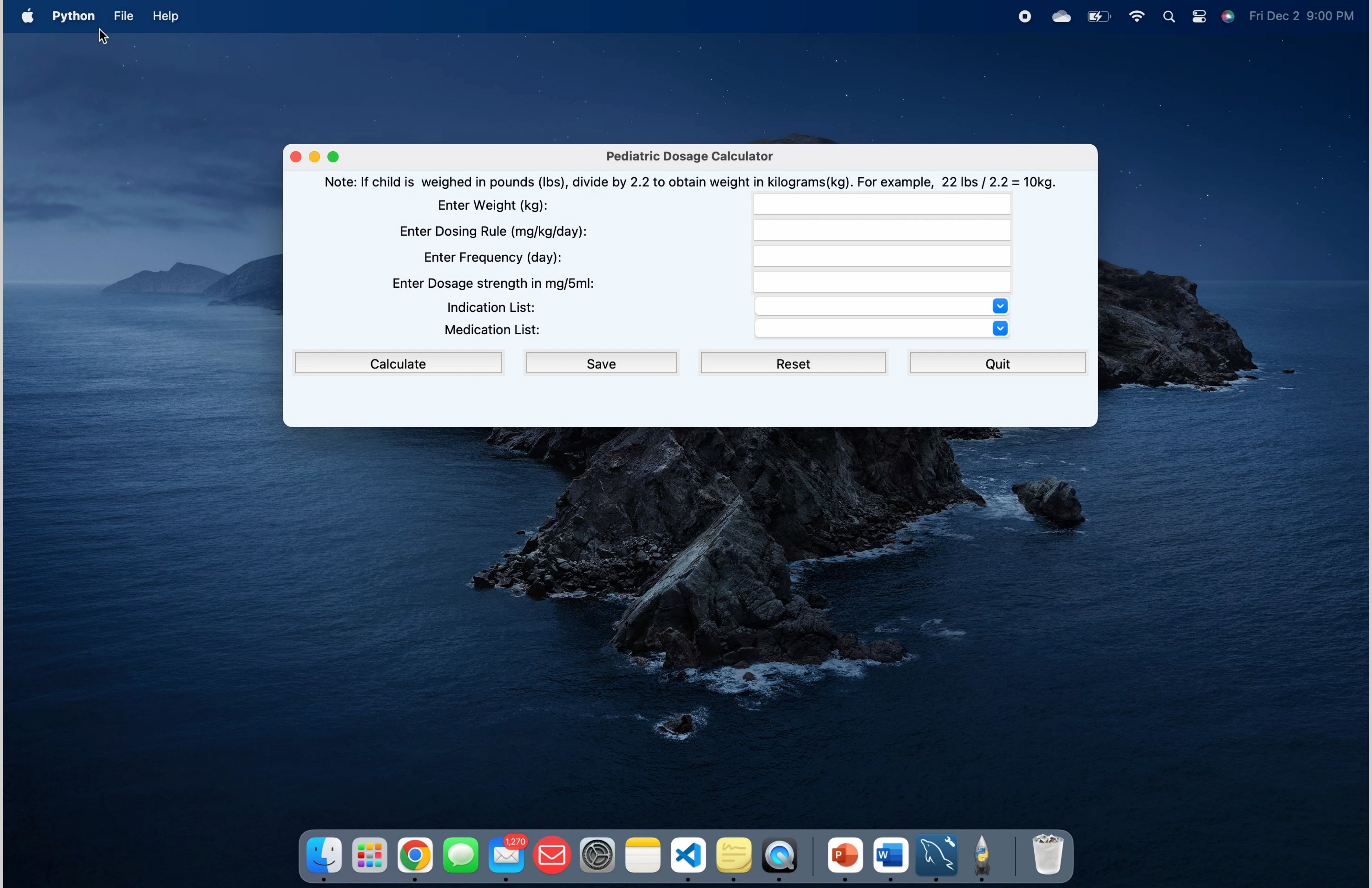
```python
        self.weight_entry = tkinter.Entry(self.top_frame, font= "bold", width = 30, justify="left")
        self.weight_entry.grid(row=1, column=1)
        self.weight_label = tkinter.Label(self.top_frame, bg= "aliceblue", font= ("bold", 14), text="Enter Weight (kg): ", justify="left")
        self.weight_label.grid(row=1, column=0)

        self.dosing_rule_entry = tkinter.Entry(self.top_frame, font= "bold", width = 30, justify="left")
        self.dosing_rule_entry.grid(row=2, column=1)
        self.dosing_rule_label = tkinter.Label(self.top_frame,bg= "aliceblue", font= ("bold", 14), text="Enter Dosing Rule (mg/kg/day): ", justify="left")
        self.dosing_rule_label.grid(row=2, column=0)

        self.frequency_entry = tkinter.Entry(self.top_frame, font= "bold", width = 30, justify="left")
        self.frequency_entry.grid(row=3, column=1)
        self.frequency_label = tkinter.Label(self.top_frame, bg= "aliceblue", font= ("bold", 14), text="Enter Frequency (day): ", justify="left")
        self.frequency_label.grid(row=3, column=0)

        self.strength_entry = tkinter.Entry(self.top_frame, font= "bold", width = 30, justify="left")
        self.strength_entry.grid(row=4, column=1)
        self.strength_label = tkinter.Label(self.top_frame, text="Enter Dosage strength in mg/5ml: ", bg= "aliceblue", font= ("bold", 14), justify="left")
        self.strength_label.grid(row=4, column=0)

        self.disease_label = tkinter.Label(self.top_frame, text= "Indication List:  ", bg= "aliceblue", width = 30, font= ("bold", 14),justify="left")
        self.disease_label.grid(row=5, column=0)
        data1 = ("Acute Otitis Media","Allergy Relief", "Pain Relief")
        self.disease_combobox = Combobox(self.top_frame, values = data1, width= 28, font= "bold")
        self.disease_combobox.grid(row=5, column=1)

        self.medication_label = tkinter.Label(self.top_frame, bg= "aliceblue", text="Medication List: ", width=30, font= ("bold", 14), justify="left")
        self.medication_label.grid(row=6, column=0)
        data2 = ("Amoxicillin", "Diphenylhramine", "Ibuprofen", "Tynelol")
        self.medication_combobox = Combobox(self.top_frame, values = data2,  font= "bold", width=28 )
        self.medication_combobox.grid(row=6, column=1)

        # Middle frame
        # create widgets for mid frame
        self.cal_button = tkinter.Button(self.mid_frame, bg= "alice blue", text = "Calculate", font= ("bold", 14),  command=self.calculateDose)
        self.cal_button.grid(row=9, column=0, columnspan=1, pady=10, padx=10, ipadx=66)

        # create store button for mid frame
        self.save_button = tkinter.Button(self.mid_frame, bg= "alice blue", font= ("bold", 14), text = "Save", command= self.save)
        self.save_button.grid(row=9, column=1, columnspan=1, pady=10, padx=10, ipadx=50)

        # create reset button for mid frame
        self.reset_button = tkinter.Button(self.mid_frame, bg= "alice blue",  font= ("bold", 14),  text = "Reset", command= self.reset)
```



```
        # call the label widget's pack method
        self.top_frame.pack()
        self.mid_frame.pack()
        self.bottom_frame.pack()

        tkinter.mainloop()

        # To reset the calculator
    def reset(self):
        conn = mysql.connector.connect(
        host= "localhost",
        user= "root",
        passwd= "password",
        database = "dosageCalculator_db")
        db_cursor = conn.cursor()

        # To clear texboxes
        self.weight_entry.delete (0, END)
        self.strength_entry.delete (0, END)
        self.frequency_entry.delete (0, END)
        self.dosing_rule_entry.delete (0, END)
        self.disease_combobox.set(" ")
        self.medication_combobox.set(" ")
        conn.commit()
        conn.close()

    #To store to DB and clear fields.
    def save(self):
        if self.weight_entry.get() == " " or self.strength_entry.get == " " or self.frequency_entry.get() == " " or self.dosing_rule_entry.get () ==" ":
            tkMessageBox.showerror('Error!', 'Enter correct details')
        else:
            conn = mysql.connector.connect(
        host= "localhost",
        user= "root",
        passwd= "password",
        database = "dosageCalculator_db")
        db_cursor = conn.cursor()

        # To insert data into a particular column
        sql_wt = "Insert into Weight_Info (weight_kg) values (%s)"
        db_cursor.execute(sql_wt, (self.weight_entry.get(),))
        conn.commit()
        sql_str = "Insert into Strength_Info (dose_strength_mg_per_5ml) values (%s)"
```



```
            frequency = db_cursor.execute(sql_fq)
            result_frequency = db_cursor.fetchall()
            for frequency in result_frequency:
                input_freq  = (float(frequency[0]))
        else:
            sql_fq = "Insert into Frequency_Info (frequency_per_day) values (%s)"

            frequency = db_cursor.execute(sql_fq, (self.frequency_entry.get(),))
            input_freq = float(self.frequency_entry.get())
            conn.commit()
            print(input_freq)
            print(type(input_freq))
            conn.commit()

        dose_rule = float(self.dosing_rule_entry.get())
        if dose_rule == "":
            sql_dr = "select dosing_rule_mg_per_kg_per_day from Dosing_Rule_Info where class='A'"
            dose_rule = db_cursor.execute(sql_dr)
            result_dose_rule = db_cursor.fetchall()
            for dose_rule in result_dose_rule:
                input_dose  = (float(dose_rule[0]))
        else:
            sql_dr = "Insert into Dosing_Rule_Info (dosing_rule_mg_per_kg_per_day) values (%s)"

            dose_rule = db_cursor.execute(sql_fq, (self.dosing_rule_entry.get(),))
            input_dose = float(self.dosing_rule_entry.get())
            conn.commit()
            print(input_dose)
            print(type(input_dose))
            conn.commit()

        mg_per_day = (input_weight * input_dose)
        mg_per_dose = mg_per_day / input_freq
        dose_to_ml = mg_per_dose / input_strength
        dose_per_ml = round( dose_to_ml,2)

        print(dose_per_ml)
        self.calculated_dose_value.set("Recommended dose is: " + str(dose_per_ml) + " mL to be taken " + str(input_freq) + " time(s) daily.")
        conn.commit()
        conn.close()

my_gui = MyGUI()
```



Pediatric Dosage Calculator

Note: If child is weighed in pounds (lbs), divide by 2.2 to obtain weight in kilograms(kg). For example, 22 lbs / 2.2 = 10kg.

Enter Weight (kg):

Enter Dosing Rule (mg/kg/day):

Enter Frequency (day):

Enter Dosage strength in mg/5ml:

Indication List:

Medication List: