SCHOOL OF INFORMATION SCIENCE COLLEGE OF COMPUTING, INFORMATICS AND MEDIA UNIVERSITI TEKNOLOGI MARA MERBOK KEDAH

DIPLOMA IN LIBRARY INFORMATIC

(CDIM 144)

PROGRAMMING FOR LIBRARIES

(IML 208)

GROUP ASSIGNMENT: KINDERGARTEN SYSTEM

PREPARED BY:

NAMA	**ID NUMBER**
AINUL MARDHIAH BINTI SHAFIE	2022661918
ATIQAH BATRISYIA BINTI ZAKARIA	2022627812
NUR SAKINAH BINTI MOHD. SAHRULNIZAM	2022485826
SITI NUR AISYAH BINTI RIDZUAN	2022814118

CLASS: KIM 1443E

PREPARED FOR:

SIR AIRUL SHAZWAN BIN NORSHAHIMI

SUBMISSION DATE: 16TH JANUARY 2024

GROUP ASSIGNMENT: KINDERGARTEN SYSTEM

PREPARED BY:

NAMA	ID NUMBER
AINUL MARDHIAH BINTI SHAFIE	2022661918
ATIQAH BATRISYIA BINTI ZAKARIA	2022627812
NUR SAKINAH BINTI MOHD. SAHRULNIZAM	2022485826
SITI NUR AISYAH BINTI RIDZUAN	2022814118

SCHOOL OF INFORMATION SCIENCE COLLEGE OF COMPUTING, INFORMATICS AND MEDIA UNIVERSITI TEKNOLOGI MARA MERBOK KEDAH

SUBMISSION DATE:

16TH JANUARY 2024

ACKNOWLEDGEMENT

Assalamualaikum warahamtullahi wabarakatuh

First and foremost, alhamdulillah all praises to Allah for the strength and His blessing for us to completing this assignment. This assignment was prepared for subject Programming for Libraries (IML 208) and alhamdulillah we got a lot of information from doing this assignment.

Special appreciation goes to our lecturer Sir Airul Shazwan Bin Norshahimi lecture for this subject who had guided, and whose encouragement helped us to go through for this assignment and without him it’s very difficult for us to finish this assignment first, it was very hard because initially we had a little knowledge about programming but after doing so many research gradually our knowledge had increased throughout completing this assignment. We are so grateful to get a chance doing this assignment with Sir Airul.

We would also want to say thank you to our lovely parents and all the other family members who give us support to finish this assignment.

Also, thanks to all our friends for their corporation, suggestion during this project until it finish.

TABLE OF CONTENT

1.0 INTRODUCTION

The project that we chose is kindergarten. Our group are focused on streamlining and enhancing key processes within the kindergarten environment through the subject of programming for libraries. Recognizing the challenges faced in the current system, our group has identified three critical areas for improvement which are the complex registration process, the difficulty teachers encounter in calculating their net salary, and the need for a more efficient method of informing students about available kindergarten subjects.

In this project, our primary objectives are threefold. Firstly, we aim to develop a systematic and accessible registration process that ensures efficiency and inclusivity. Secondly, we strive to simplify the process for teachers to calculate their net salary from the gross amount received. Lastly, we endeavor to provide students with a user-friendly platform to explore and register for available kindergarten subjects.

To achieve these goals, our group will employ a comprehensive approach, integrating coding, flowchart creation, and database implementation. The coding aspect will involve developing solutions for teacher registration, student registration, and subject registration, while flowcharts will visually represent the logical processes involved. The database implementation will contribute to the seamless management of information and data associated with these crucial aspects of kindergarten administration. Throughout this project, we have learned to create a more seamless and user-friendly experience for teachers, students, and administrators.

2.0 PROBLEM STATEMENT

1. Difficult registration process.

The current student registration process is difficult to use and insert the insert the information accurately. There also lack of validation check during registration, leading to potential error and incomplete submissions.

2. The teacher having difficulty in knowing their net salary.

There is no function in the system that makes it easy for teachers to view and calculate their net salary. It might be difficult for teachers to get clear and simple information about their total net income, bonuses, and deductions.

3. To be inform about the kindergarten subject availability to the student.

There is no system in place to notify students about the subjects that are taught in the kindergarten. Students are not provided with details on the subjects offered, making it challenging for them to plan their courses effectively.

3.0 OBJECTIVES

1. Systematic and Accessible Registration Process

   Enhance the registration process to be systematic, user-friendly, and accessible for all stakeholders involved.

2. Teacher Net Salary Calculation

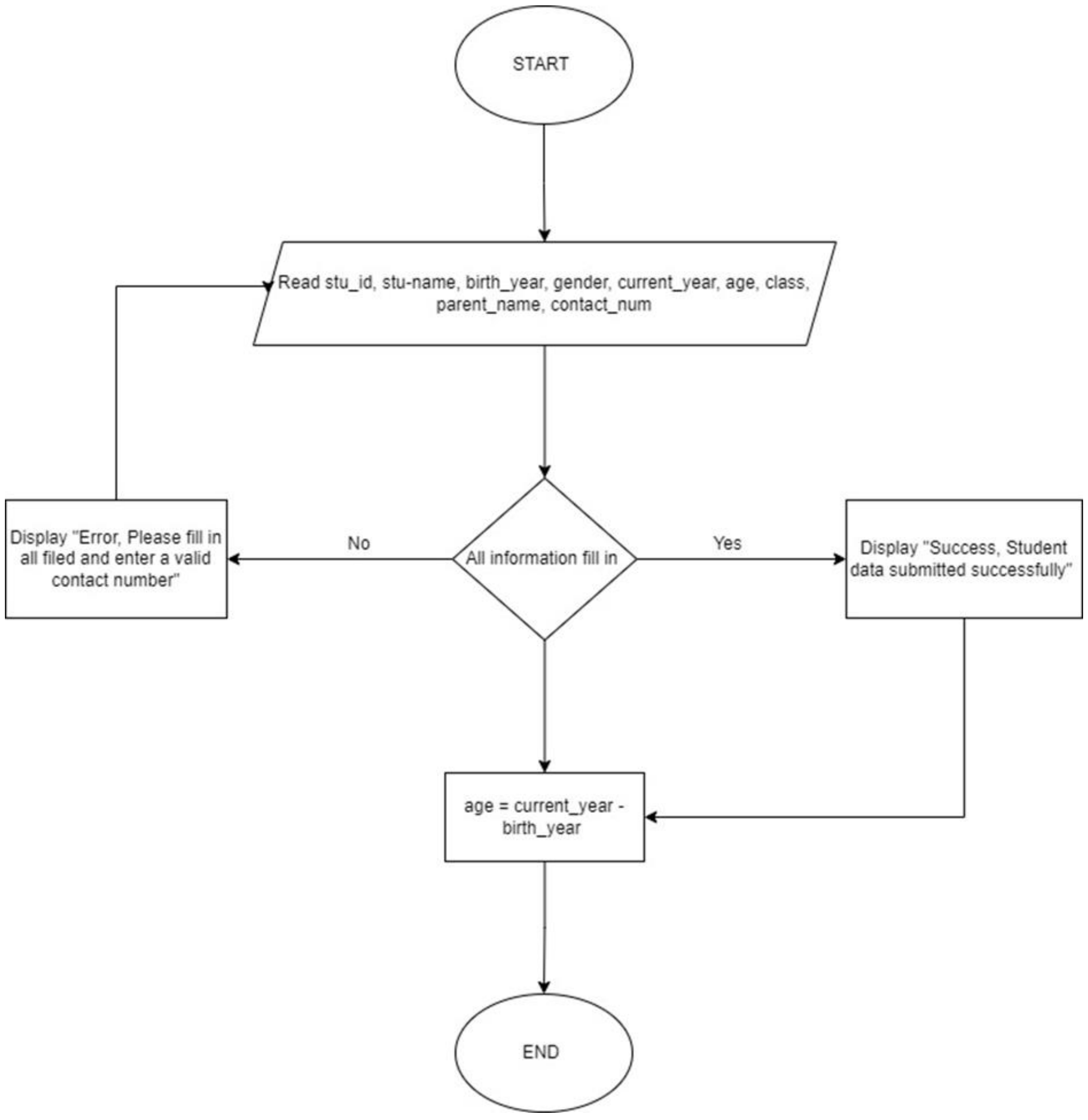
   Simplify the process for teachers to calculate their net salary accurately based on their gross salary.

3. Subject Registration Option for Students

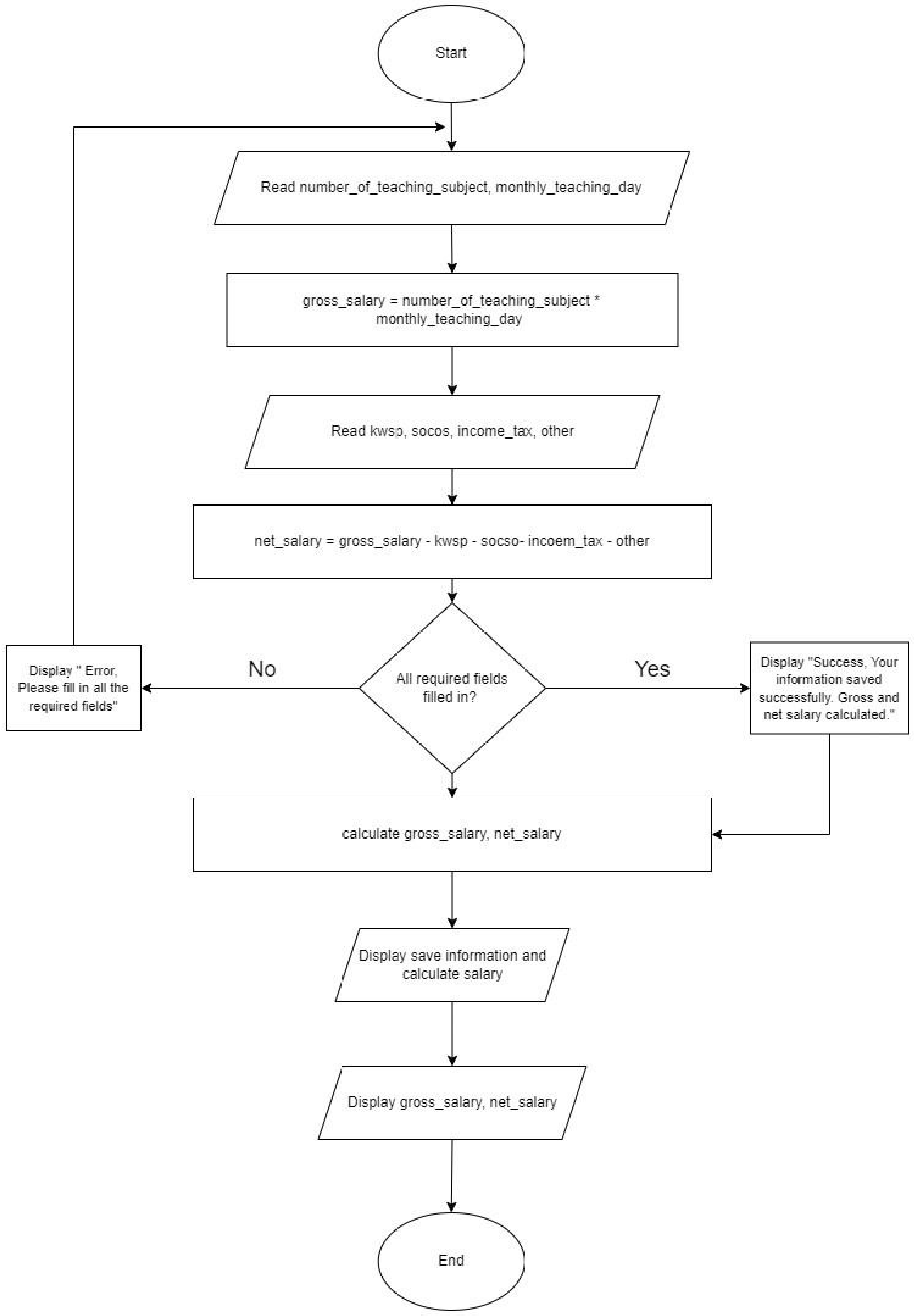
   Empower students with the ability to register for available subjects in kindergarten.

4.0 FLOWCHART

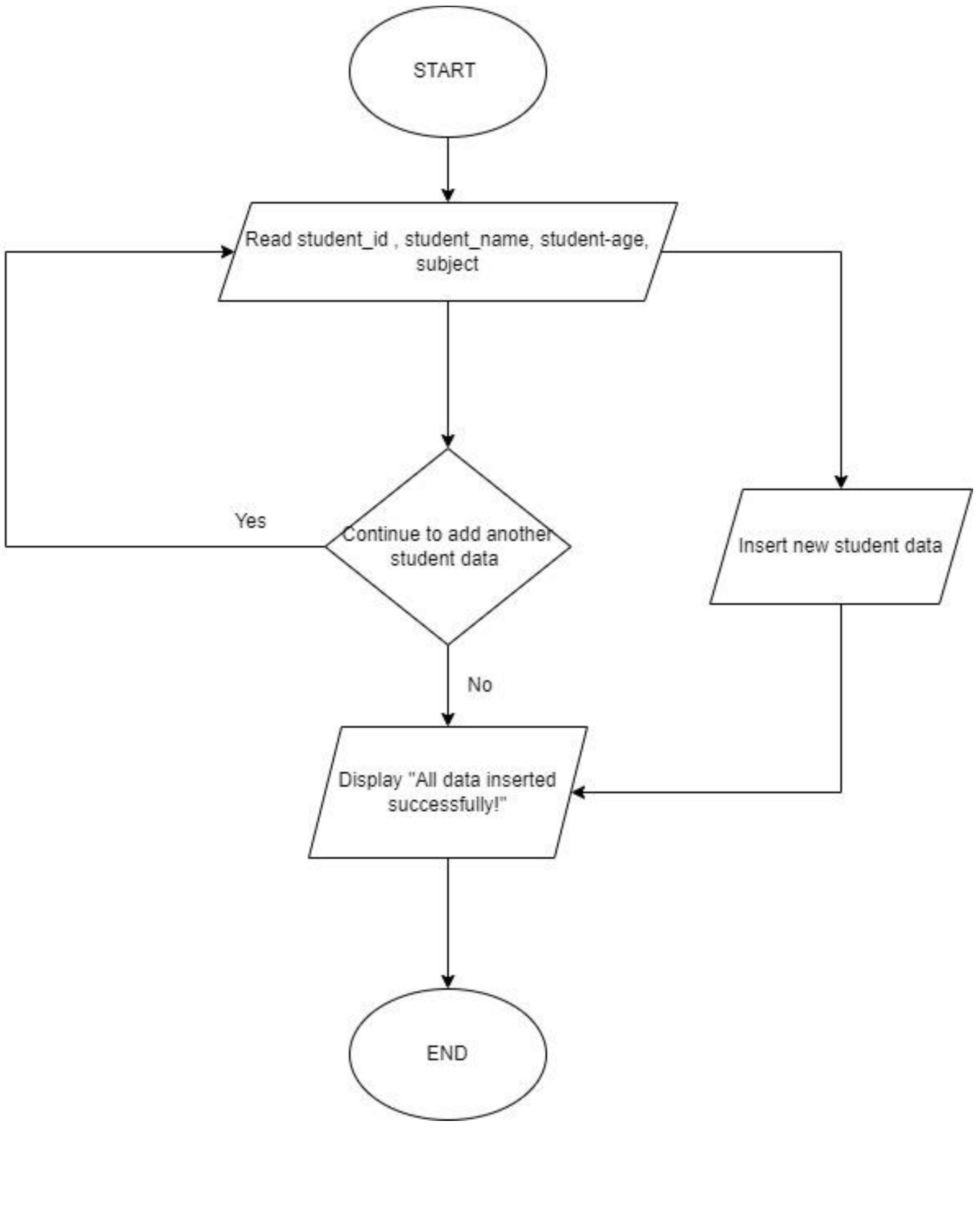
4.1 Student Registration


START
Read stu_id, stu-name, birth_year, gender, current_year, age, class, parent_name, contact_num
All information fill in
No
Yes
Display "Error, Please fill in all filed and enter a valid contact number"
Display "Success, Student data submitted successfully"
age = current_year - birth_year
END

4.2 Teacher Calculation

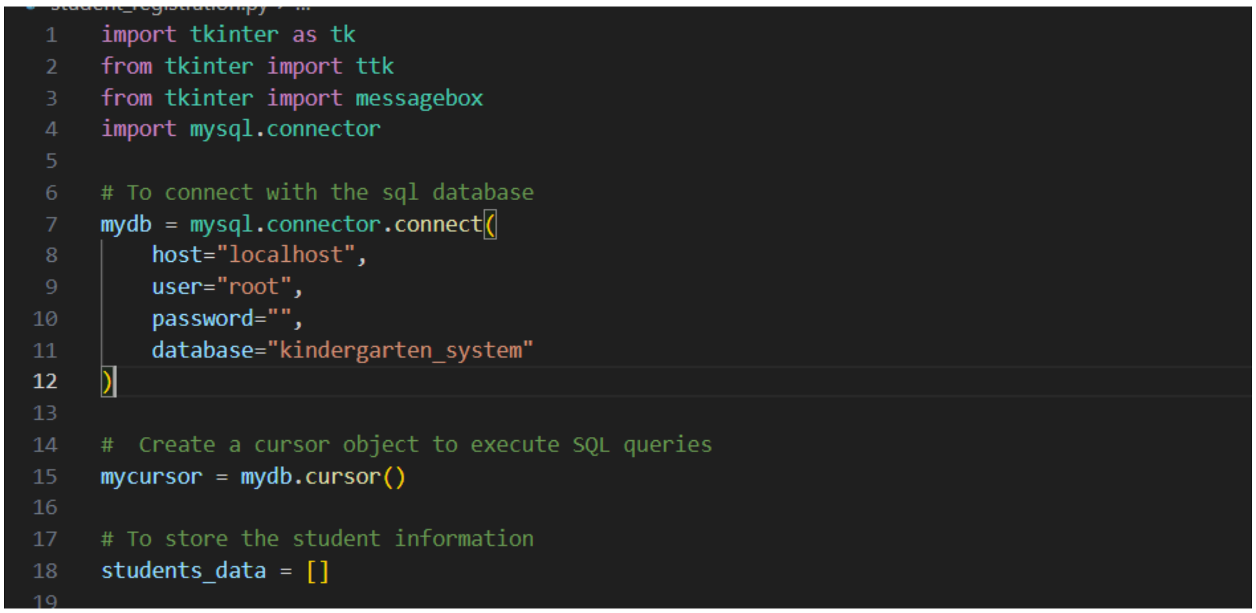

Start
Read number_of_teaching_subject, monthly_teaching_day
gross_salary = number_of_teaching_subject * monthly_teaching_day
Read kwsp, socos, income_tax, other
net_salary = gross_salary - kwsp - socso- incoem_tax - other
All required fields filled in?
No
Yes
Display " Error, Please fill in all the required fields"
Display "Success, Your information saved successfully. Gross and net salary calculated."
calculate gross_salary, net_salary
Display save information and calculate salary
Display gross_salary, net_salary
End

4.3 Subject Registration

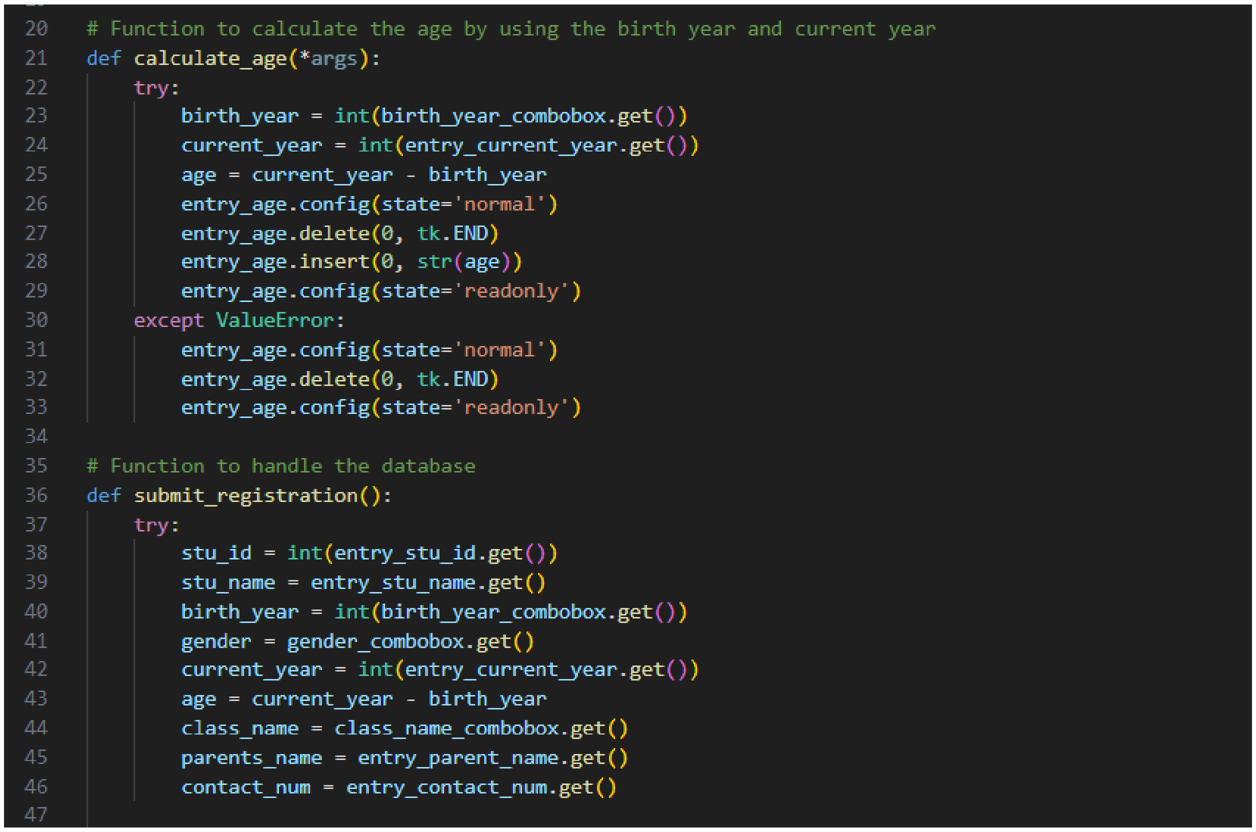

START
Read student_id , student_name, student-age, subject
Yes
Continue to add another student data
Insert new student data
No
Display "All data inserted successfully!"
END

5.0 SHAPSHOT OF CODING

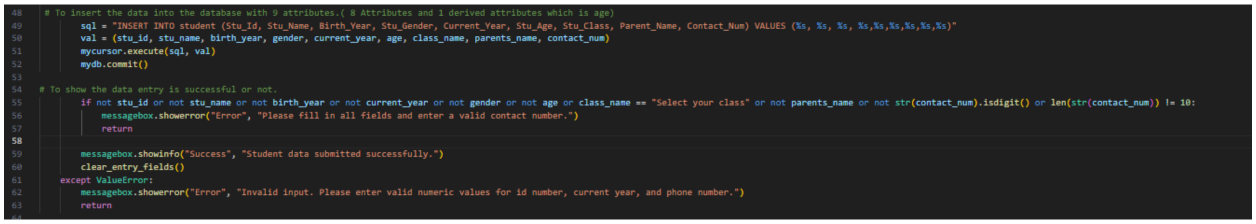
5.1 Student Registration



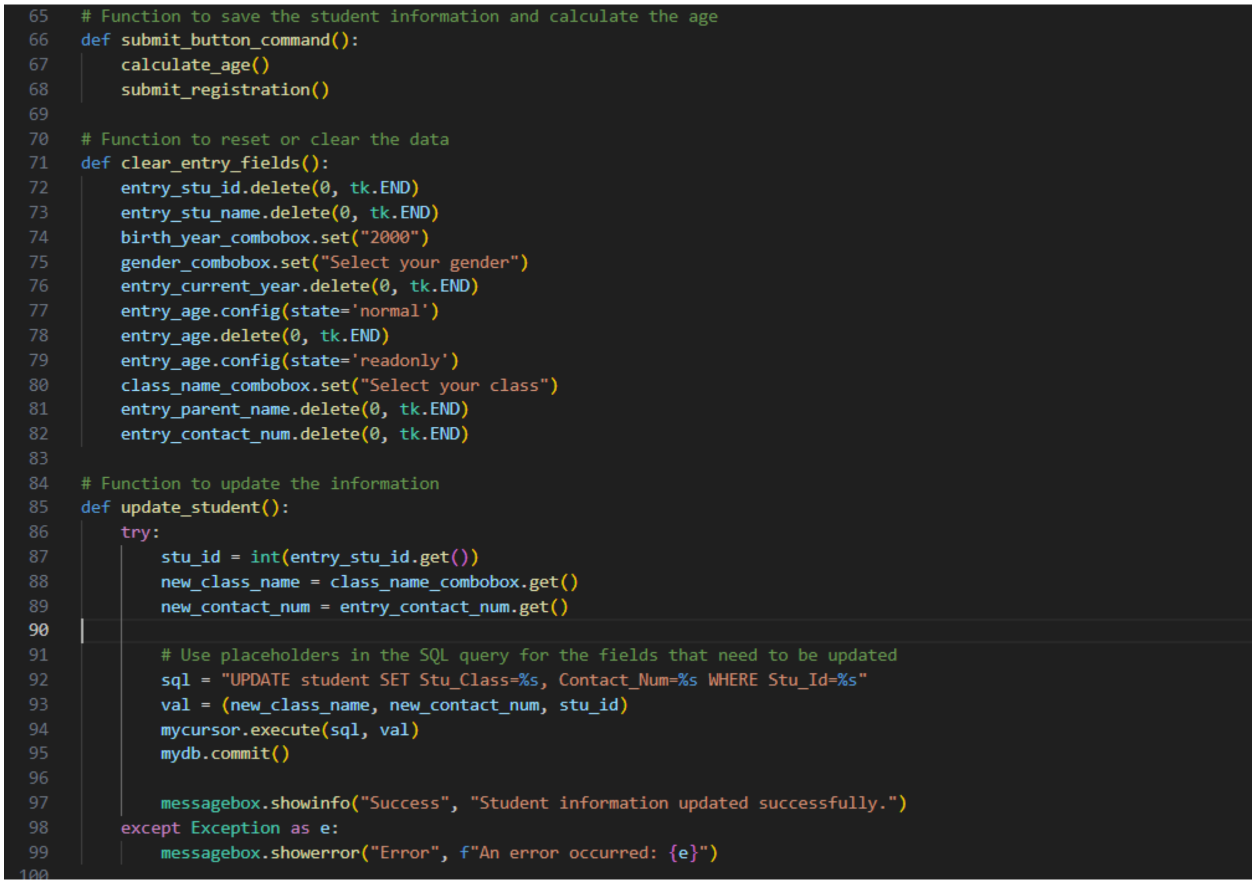
```python
import tkinter as tk
from tkinter import ttk
from tkinter import messagebox
import mysql.connector

# To connect with the sql database
mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="kindergarten_system"
)

#  Create a cursor object to execute SQL queries
mycursor = mydb.cursor()

# To store the student information
students_data = []

```



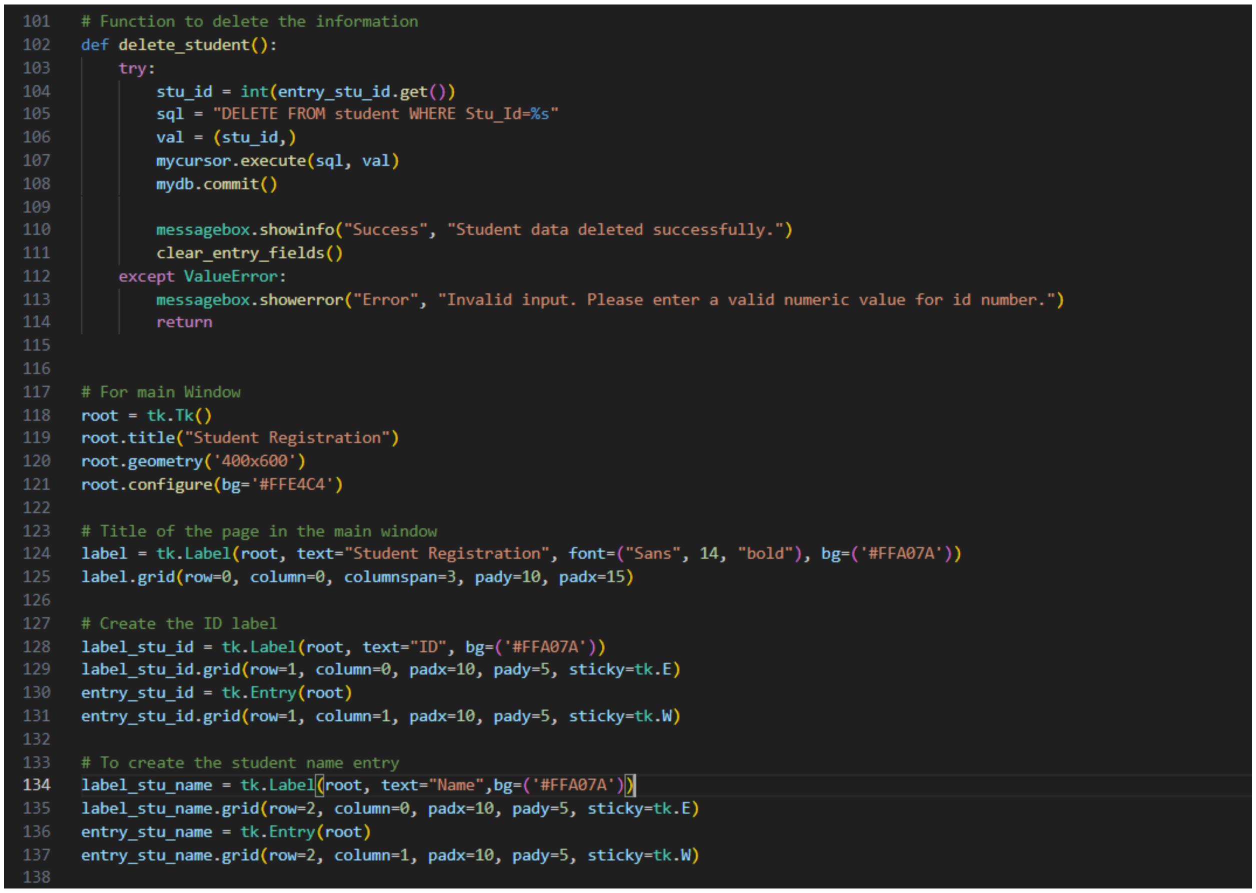
```python
# Function to calculate the age by using the birth year and current year
def calculate_age(*args):
    try:
        birth_year = int(birth_year_combobox.get())
        current_year = int(entry_current_year.get())
        age = current_year - birth_year
        entry_age.config(state='normal')
        entry_age.delete(0, tk.END)
        entry_age.insert(0, str(age))
        entry_age.config(state='readonly')
    except ValueError:
        entry_age.config(state='normal')
        entry_age.delete(0, tk.END)
        entry_age.config(state='readonly')

# Function to handle the database
def submit_registration():
    try:
        stu_id = int(entry_stu_id.get())
        stu_name = entry_stu_name.get()
        birth_year = int(birth_year_combobox.get())
        gender = gender_combobox.get()
        current_year = int(entry_current_year.get())
        age = current_year - birth_year
        class_name = class_name_combobox.get()
        parents_name = entry_parent_name.get()
        contact_num = entry_contact_num.get()

```



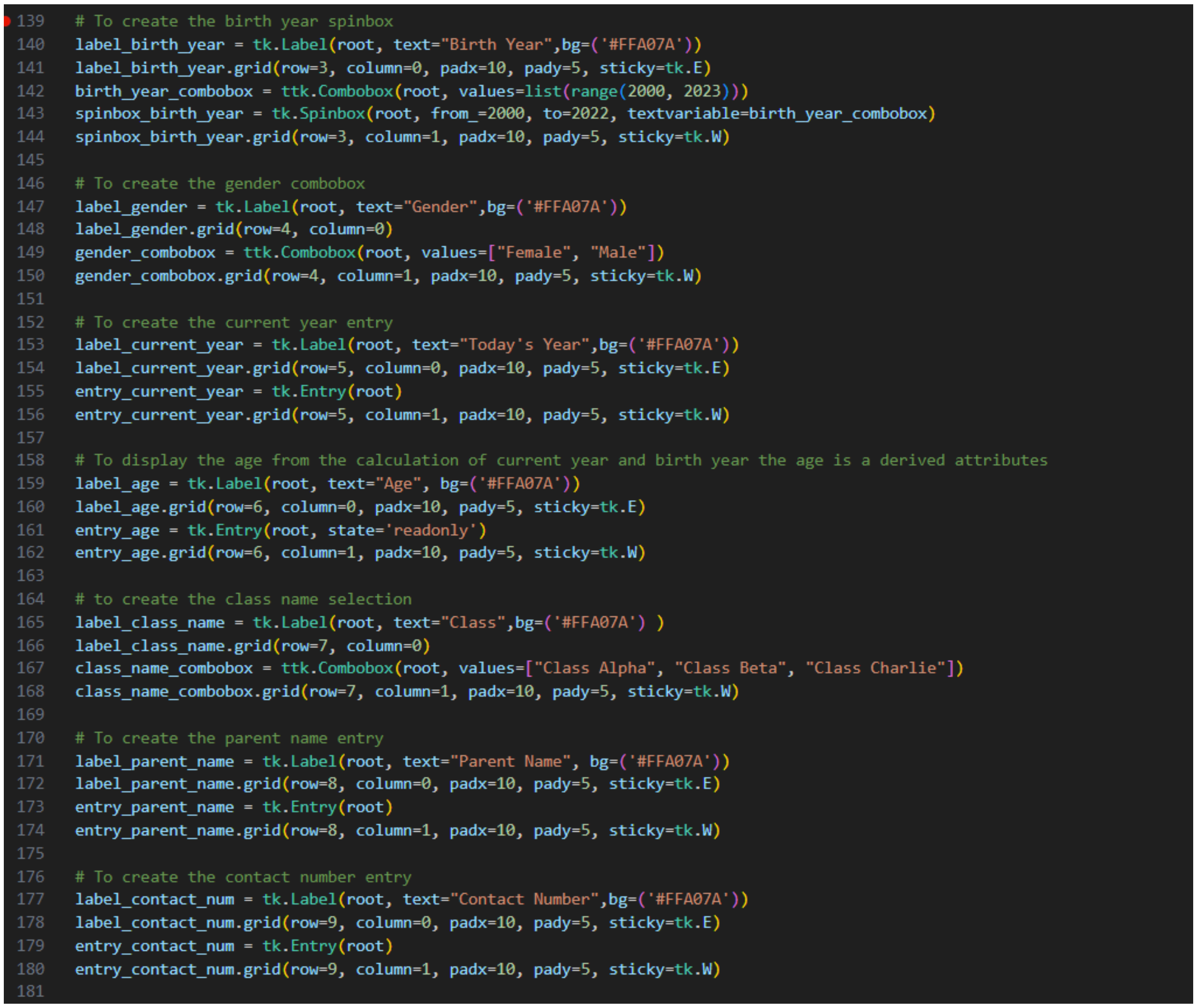
```
    # To insert the data into the database with 9 attributes.( 8 Attributes and 1 derived attributes which is age)
        sql = "INSERT INTO student (Stu_Id, Stu_Name, Birth_Year, Stu_Gender, Current_Year, Stu_Age, Stu_Class, Parent_Name, Contact_Num) VALUES (%s, %s, %s, %s,%s,%s,%s,%s,%s)"
        val = (stu_id, stu_name, birth_year, gender, current_year, age, class_name, parents_name, contact_num)
        mycursor.execute(sql, val)
        mydb.commit()

    # To show the data entry is successful or not.
        if not stu_id or not stu_name or not birth_year or not current_year or not gender or not age or class_name == "Select your class" or not parents_name or not str(contact_num).isdigit() or len(str(contact_num)) != 10:
            messagebox.showerror("Error", "Please fill in all fields and enter a valid contact number.")
            return

        messagebox.showinfo("Success", "Student data submitted successfully.")
        clear_entry_fields()
    except ValueError:
        messagebox.showerror("Error", "Invalid input. Please enter valid numeric values for id number, current year, and phone number.")
        return
```



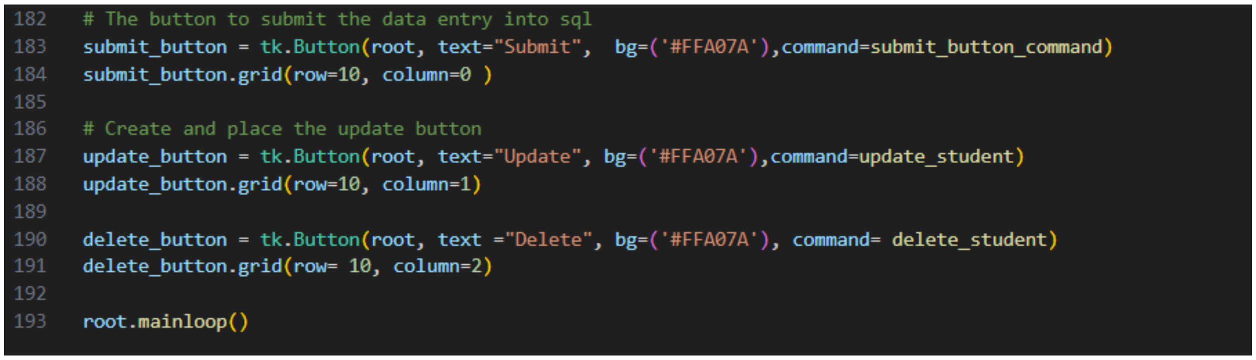
```
# Function to save the student information and calculate the age
def submit_button_command():
    calculate_age()
    submit_registration()

# Function to reset or clear the data
def clear_entry_fields():
    entry_stu_id.delete(0, tk.END)
    entry_stu_name.delete(0, tk.END)
    birth_year_combobox.set("2000")
    gender_combobox.set("Select your gender")
    entry_current_year.delete(0, tk.END)
    entry_age.config(state='normal')
    entry_age.delete(0, tk.END)
    entry_age.config(state='readonly')
    class_name_combobox.set("Select your class")
    entry_parent_name.delete(0, tk.END)
    entry_contact_num.delete(0, tk.END)

# Function to update the information
def update_student():
    try:
        stu_id = int(entry_stu_id.get())
        new_class_name = class_name_combobox.get()
        new_contact_num = entry_contact_num.get()

        # Use placeholders in the SQL query for the fields that need to be updated
        sql = "UPDATE student SET Stu_Class=%s, Contact_Num=%s WHERE Stu_Id=%s"
        val = (new_class_name, new_contact_num, stu_id)
        mycursor.execute(sql, val)
        mydb.commit()

        messagebox.showinfo("Success", "Student information updated successfully.")
    except Exception as e:
        messagebox.showerror("Error", f"An error occurred: {e}")
```



```
# Function to delete the information
def delete_student():
    try:
        stu_id = int(entry_stu_id.get())
        sql = "DELETE FROM student WHERE Stu_Id=%s"
        val = (stu_id,)
        mycursor.execute(sql, val)
        mydb.commit()

        messagebox.showinfo("Success", "Student data deleted successfully.")
        clear_entry_fields()
    except ValueError:
        messagebox.showerror("Error", "Invalid input. Please enter a valid numeric value for id number.")
        return


# For main Window
root = tk.Tk()
root.title("Student Registration")
root.geometry('400x600')
root.configure(bg='#FFE4C4')

# Title of the page in the main window
label = tk.Label(root, text="Student Registration", font=("Sans", 14, "bold"), bg=('#FFA07A'))
label.grid(row=0, column=0, columnspan=3, pady=10, padx=15)

# Create the ID label
label_stu_id = tk.Label(root, text="ID", bg=('#FFA07A'))
label_stu_id.grid(row=1, column=0, padx=10, pady=5, sticky=tk.E)
entry_stu_id = tk.Entry(root)
entry_stu_id.grid(row=1, column=1, padx=10, pady=5, sticky=tk.W)

# To create the student name entry
label_stu_name = tk.Label(root, text="Name",bg=('#FFA07A'))
label_stu_name.grid(row=2, column=0, padx=10, pady=5, sticky=tk.E)
entry_stu_name = tk.Entry(root)
entry_stu_name.grid(row=2, column=1, padx=10, pady=5, sticky=tk.W)
```

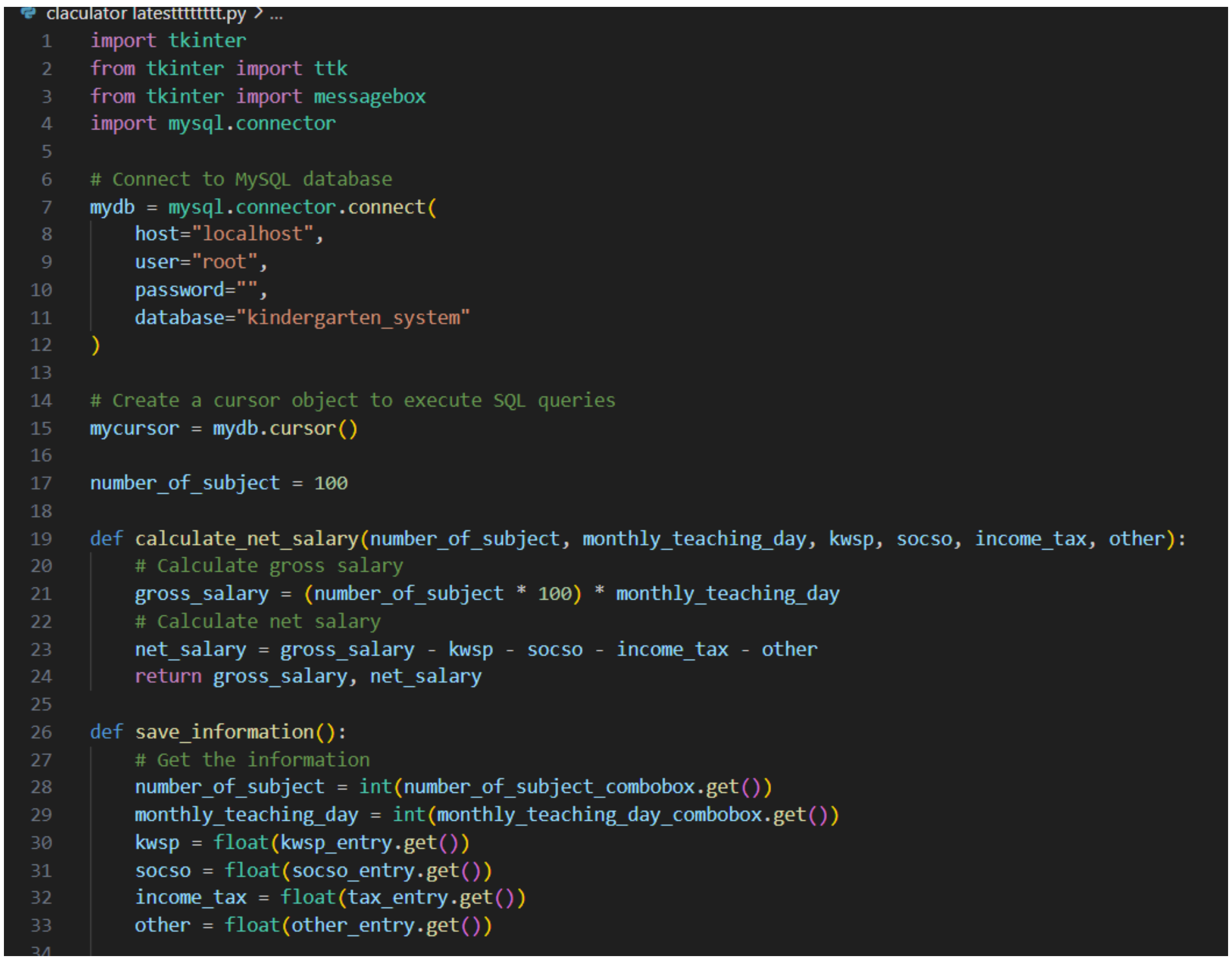


```python
# To create the birth year spinbox
label_birth_year = tk.Label(root, text="Birth Year",bg=('#FFA07A'))
label_birth_year.grid(row=3, column=0, padx=10, pady=5, sticky=tk.E)
birth_year_combobox = ttk.Combobox(root, values=list(range(2000, 2023)))
spinbox_birth_year = tk.Spinbox(root, from_=2000, to=2022, textvariable=birth_year_combobox)
spinbox_birth_year.grid(row=3, column=1, padx=10, pady=5, sticky=tk.W)

# To create the gender combobox
label_gender = tk.Label(root, text="Gender",bg=('#FFA07A'))
label_gender.grid(row=4, column=0)
gender_combobox = ttk.Combobox(root, values=["Female", "Male"])
gender_combobox.grid(row=4, column=1, padx=10, pady=5, sticky=tk.W)

# To create the current year entry
label_current_year = tk.Label(root, text="Today's Year",bg=('#FFA07A'))
label_current_year.grid(row=5, column=0, padx=10, pady=5, sticky=tk.E)
entry_current_year = tk.Entry(root)
entry_current_year.grid(row=5, column=1, padx=10, pady=5, sticky=tk.W)

# To display the age from the calculation of current year and birth year the age is a derived attributes
label_age = tk.Label(root, text="Age", bg=('#FFA07A'))
label_age.grid(row=6, column=0, padx=10, pady=5, sticky=tk.E)
entry_age = tk.Entry(root, state='readonly')
entry_age.grid(row=6, column=1, padx=10, pady=5, sticky=tk.W)

# to create the class name selection
label_class_name = tk.Label(root, text="Class",bg=('#FFA07A') )
label_class_name.grid(row=7, column=0)
class_name_combobox = ttk.Combobox(root, values=["Class Alpha", "Class Beta", "Class Charlie"])
class_name_combobox.grid(row=7, column=1, padx=10, pady=5, sticky=tk.W)

# To create the parent name entry
label_parent_name = tk.Label(root, text="Parent Name", bg=('#FFA07A'))
label_parent_name.grid(row=8, column=0, padx=10, pady=5, sticky=tk.E)
entry_parent_name = tk.Entry(root)
entry_parent_name.grid(row=8, column=1, padx=10, pady=5, sticky=tk.W)

# To create the contact number entry
label_contact_num = tk.Label(root, text="Contact Number",bg=('#FFA07A'))
label_contact_num.grid(row=9, column=0, padx=10, pady=5, sticky=tk.E)
entry_contact_num = tk.Entry(root)
entry_contact_num.grid(row=9, column=1, padx=10, pady=5, sticky=tk.W)

```

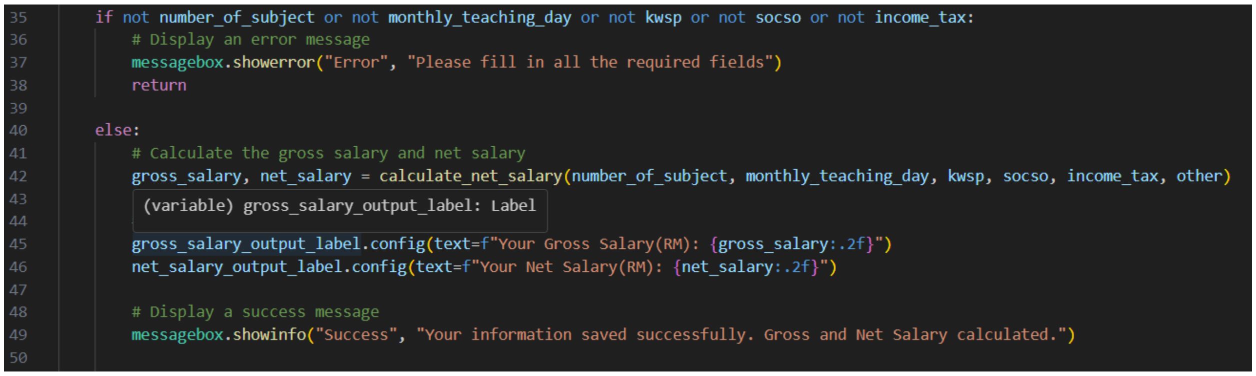


```python
# The button to submit the data entry into sql
submit_button = tk.Button(root, text="Submit",  bg=('#FFA07A'),command=submit_button_command)
submit_button.grid(row=10, column=0 )

# Create and place the update button
update_button = tk.Button(root, text="Update", bg=('#FFA07A'),command=update_student)
update_button.grid(row=10, column=1)

delete_button = tk.Button(root, text ="Delete", bg=('#FFA07A'), command= delete_student)
delete_button.grid(row= 10, column=2)

root.mainloop()
```

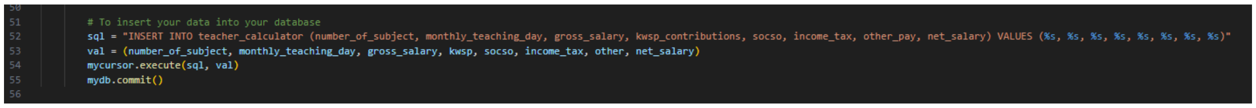
5.2 Teacher Salary Calculator


claculator latesttttttt.py > ...

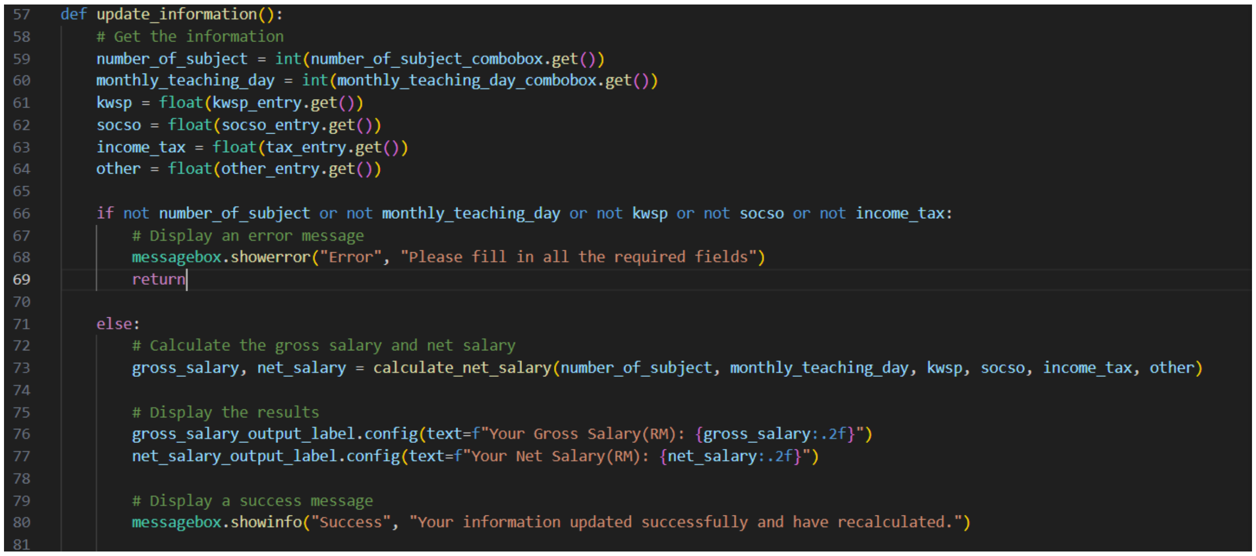
```python
import tkinter
from tkinter import ttk
from tkinter import messagebox
import mysql.connector

# Connect to MySQL database
mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="kindergarten_system"
)

# Create a cursor object to execute SQL queries
mycursor = mydb.cursor()

number_of_subject = 100

def calculate_net_salary(number_of_subject, monthly_teaching_day, kwsp, socso, income_tax, other):
    # Calculate gross salary
    gross_salary = (number_of_subject * 100) * monthly_teaching_day
    # Calculate net salary
    net_salary = gross_salary - kwsp - socso - income_tax - other
    return gross_salary, net_salary

def save_information():
    # Get the information
    number_of_subject = int(number_of_subject_combobox.get())
    monthly_teaching_day = int(monthly_teaching_day_combobox.get())
    kwsp = float(kwsp_entry.get())
    socso = float(socso_entry.get())
    income_tax = float(tax_entry.get())
    other = float(other_entry.get())
```



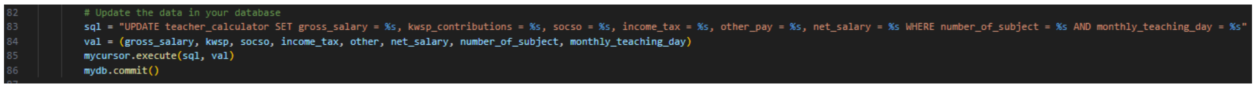
```python
    if not number_of_subject or not monthly_teaching_day or not kwsp or not socso or not income_tax:
        # Display an error message
        messagebox.showerror("Error", "Please fill in all the required fields")
        return

    else:
        # Calculate the gross salary and net salary
        gross_salary, net_salary = calculate_net_salary(number_of_subject, monthly_teaching_day, kwsp, socso, income_tax, other)
         (variable) gross_salary_output_label: Label

        gross_salary_output_label.config(text=f"Your Gross Salary(RM): {gross_salary:.2f}")
        net_salary_output_label.config(text=f"Your Net Salary(RM): {net_salary:.2f}")

        # Display a success message
        messagebox.showinfo("Success", "Your information saved successfully. Gross and Net Salary calculated.")

```



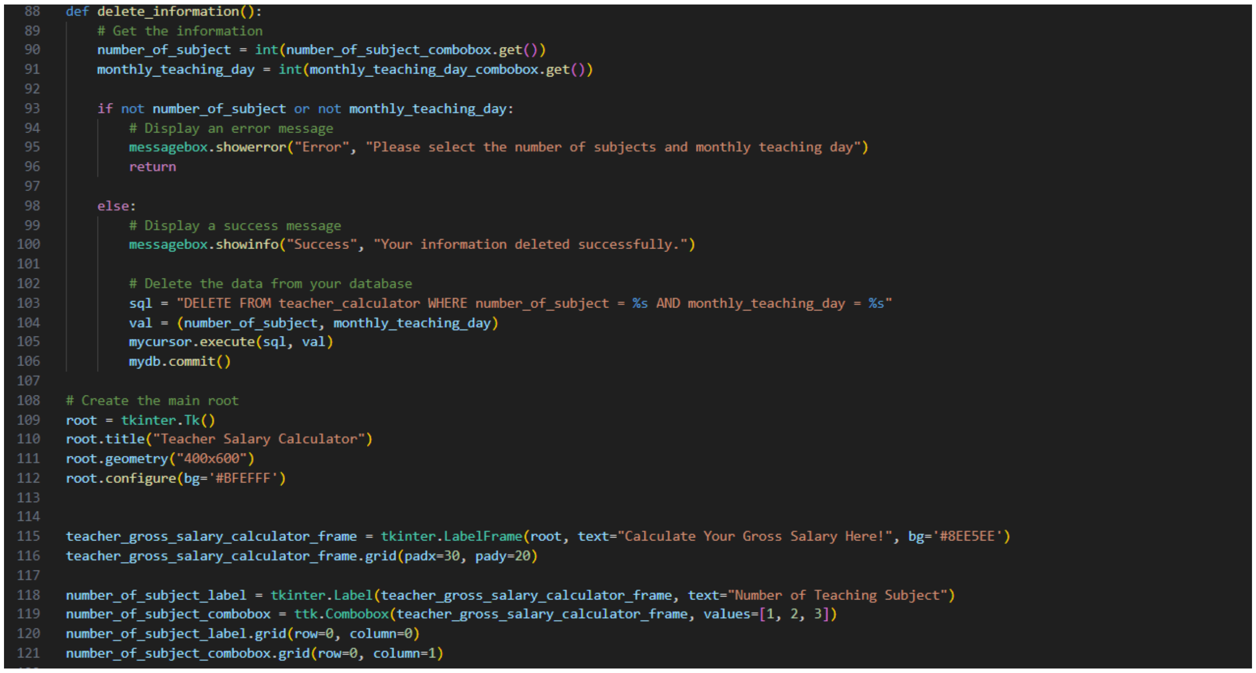
```python
        # To insert your data into your database
        sql = "INSERT INTO teacher_calculator (number_of_subject, monthly_teaching_day, gross_salary, kwsp_contributions, socso, income_tax, other_pay, net_salary) VALUES (%s, %s, %s, %s, %s, %s, %s, %s)"
        val = (number_of_subject, monthly_teaching_day, gross_salary, kwsp, socso, income_tax, other, net_salary)
        mycursor.execute(sql, val)
        mydb.commit()

```



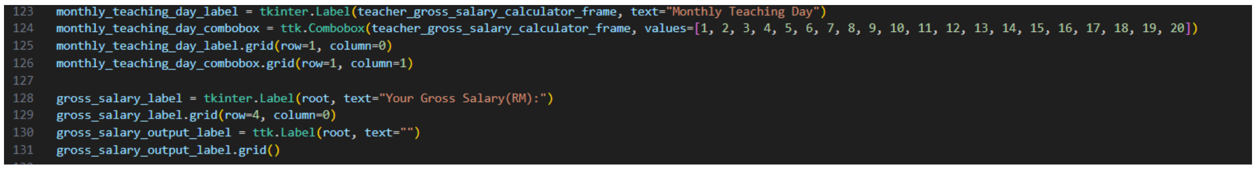
```python
def update_information():
    # Get the information
    number_of_subject = int(number_of_subject_combobox.get())
    monthly_teaching_day = int(monthly_teaching_day_combobox.get())
    kwsp = float(kwsp_entry.get())
    socso = float(socso_entry.get())
    income_tax = float(tax_entry.get())
    other = float(other_entry.get())

    if not number_of_subject or not monthly_teaching_day or not kwsp or not socso or not income_tax:
        # Display an error message
        messagebox.showerror("Error", "Please fill in all the required fields")
        return

    else:
        # Calculate the gross salary and net salary
        gross_salary, net_salary = calculate_net_salary(number_of_subject, monthly_teaching_day, kwsp, socso, income_tax, other)

        # Display the results
        gross_salary_output_label.config(text=f"Your Gross Salary(RM): {gross_salary:.2f}")
        net_salary_output_label.config(text=f"Your Net Salary(RM): {net_salary:.2f}")

        # Display a success message
        messagebox.showinfo("Success", "Your information updated successfully and have recalculated.")

```



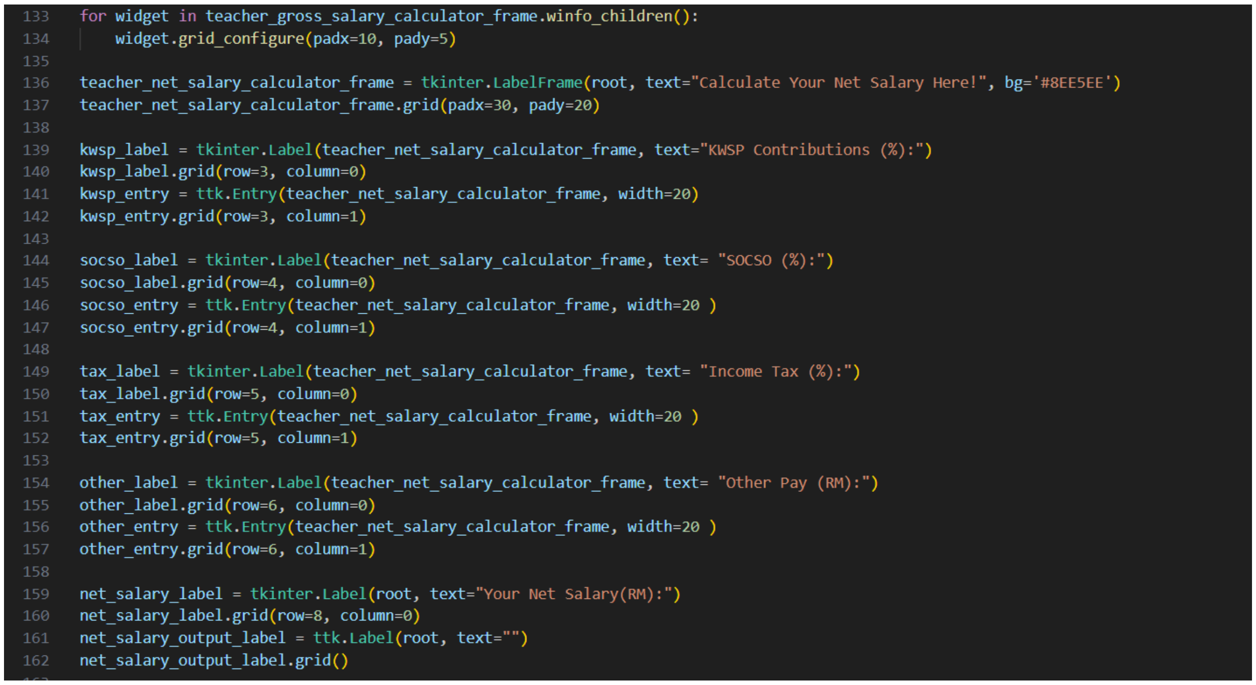
```python
        # Update the data in your database
        sql = "UPDATE teacher_calculator SET gross_salary = %s, kwsp_contributions = %s, socso = %s, income_tax = %s, other_pay = %s, net_salary = %s WHERE number_of_subject = %s AND monthly_teaching_day = %s"
        val = (gross_salary, kwsp, socso, income_tax, other, net_salary, number_of_subject, monthly_teaching_day)
        mycursor.execute(sql, val)
        mydb.commit()
```



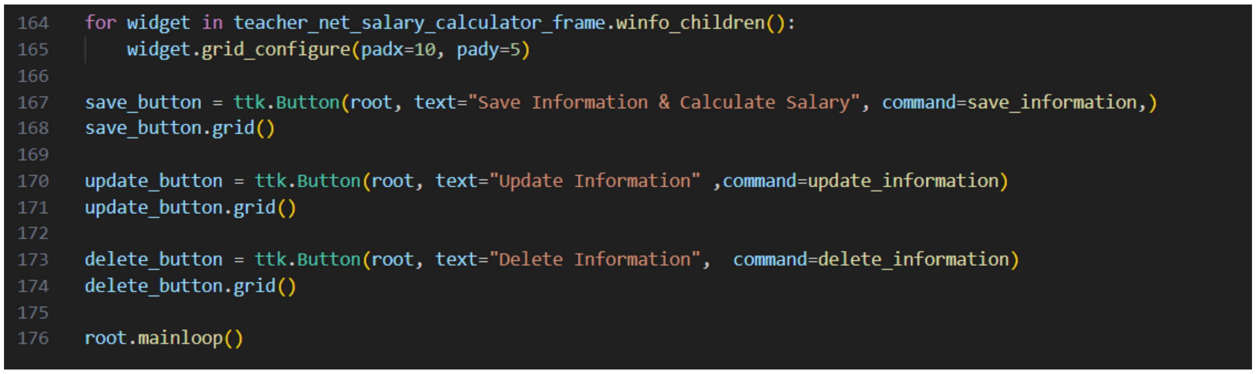
```python
def delete_information():
    # Get the information
    number_of_subject = int(number_of_subject_combobox.get())
    monthly_teaching_day = int(monthly_teaching_day_combobox.get())

    if not number_of_subject or not monthly_teaching_day:
        # Display an error message
        messagebox.showerror("Error", "Please select the number of subjects and monthly teaching day")
        return

    else:
        # Display a success message
        messagebox.showinfo("Success", "Your information deleted successfully.")

        # Delete the data from your database
        sql = "DELETE FROM teacher_calculator WHERE number_of_subject = %s AND monthly_teaching_day = %s"
        val = (number_of_subject, monthly_teaching_day)
        mycursor.execute(sql, val)
        mydb.commit()

# Create the main root
root = tkinter.Tk()
root.title("Teacher Salary Calculator")
root.geometry("400x600")
root.configure(bg='#BFEFFF')


teacher_gross_salary_calculator_frame = tkinter.LabelFrame(root, text="Calculate Your Gross Salary Here!", bg='#8EE5EE')
teacher_gross_salary_calculator_frame.grid(padx=30, pady=20)

number_of_subject_label = tkinter.Label(teacher_gross_salary_calculator_frame, text="Number of Teaching Subject")
number_of_subject_combobox = ttk.Combobox(teacher_gross_salary_calculator_frame, values=[1, 2, 3])
number_of_subject_label.grid(row=0, column=0)
number_of_subject_combobox.grid(row=0, column=1)
```



```python
monthly_teaching_day_label = tkinter.Label(teacher_gross_salary_calculator_frame, text="Monthly Teaching Day")
monthly_teaching_day_combobox = ttk.Combobox(teacher_gross_salary_calculator_frame, values=[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20])
monthly_teaching_day_label.grid(row=1, column=0)
monthly_teaching_day_combobox.grid(row=1, column=1)

gross_salary_label = tkinter.Label(root, text="Your Gross Salary(RM):")
gross_salary_label.grid(row=4, column=0)
gross_salary_output_label = ttk.Label(root, text="")
gross_salary_output_label.grid()
```

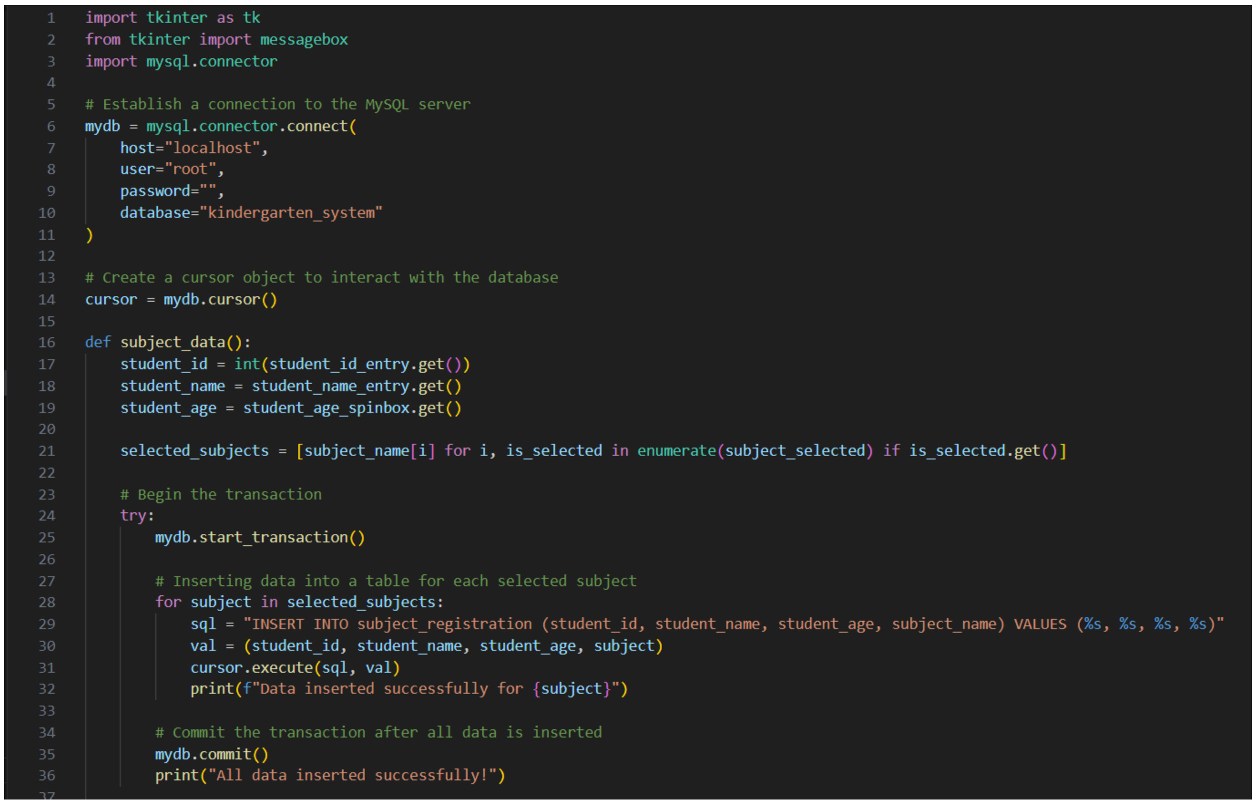


```python
for widget in teacher_gross_salary_calculator_frame.winfo_children():
    widget.grid_configure(padx=10, pady=5)

teacher_net_salary_calculator_frame = tkinter.LabelFrame(root, text="Calculate Your Net Salary Here!", bg='#8EE5EE')
teacher_net_salary_calculator_frame.grid(padx=30, pady=20)

kwsp_label = tkinter.Label(teacher_net_salary_calculator_frame, text="KWSP Contributions (%):")
kwsp_label.grid(row=3, column=0)
kwsp_entry = ttk.Entry(teacher_net_salary_calculator_frame, width=20)
kwsp_entry.grid(row=3, column=1)

socso_label = tkinter.Label(teacher_net_salary_calculator_frame, text= "SOCSO (%):")
socso_label.grid(row=4, column=0)
socso_entry = ttk.Entry(teacher_net_salary_calculator_frame, width=20 )
socso_entry.grid(row=4, column=1)

tax_label = tkinter.Label(teacher_net_salary_calculator_frame, text= "Income Tax (%):")
tax_label.grid(row=5, column=0)
tax_entry = ttk.Entry(teacher_net_salary_calculator_frame, width=20 )
tax_entry.grid(row=5, column=1)

other_label = tkinter.Label(teacher_net_salary_calculator_frame, text= "Other Pay (RM):")
other_label.grid(row=6, column=0)
other_entry = ttk.Entry(teacher_net_salary_calculator_frame, width=20 )
other_entry.grid(row=6, column=1)

net_salary_label = tkinter.Label(root, text="Your Net Salary(RM):")
net_salary_label.grid(row=8, column=0)
net_salary_output_label = ttk.Label(root, text="")
net_salary_output_label.grid()
```

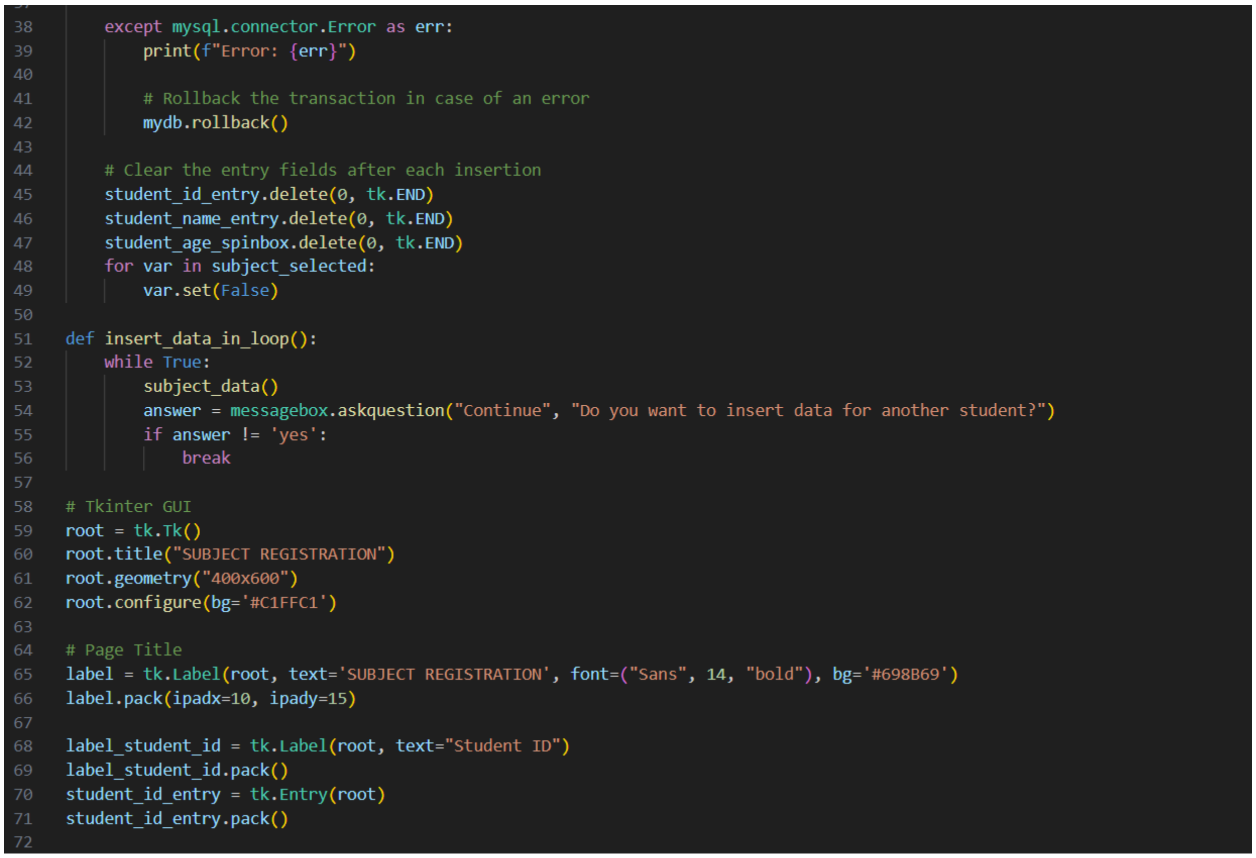


```python
for widget in teacher_net_salary_calculator_frame.winfo_children():
    widget.grid_configure(padx=10, pady=5)

save_button = ttk.Button(root, text="Save Information & Calculate Salary", command=save_information,)
save_button.grid()

update_button = ttk.Button(root, text="Update Information" ,command=update_information)
update_button.grid()

delete_button = ttk.Button(root, text="Delete Information",  command=delete_information)
delete_button.grid()

root.mainloop()
```

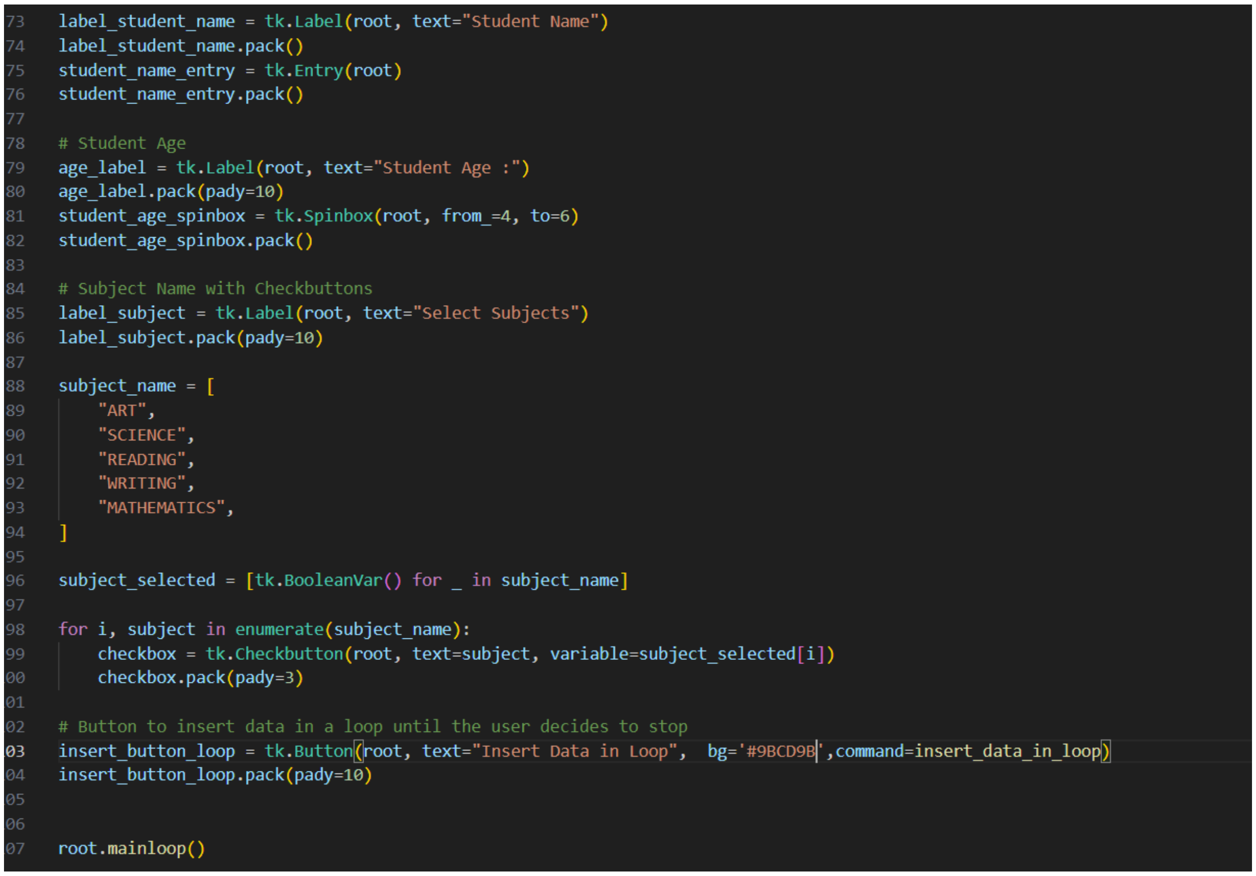
5.3 Subject Registration



```python
import tkinter as tk
from tkinter import messagebox
import mysql.connector

# Establish a connection to the MySQL server
mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="kindergarten_system"
)

# Create a cursor object to interact with the database
cursor = mydb.cursor()

def subject_data():
    student_id = int(student_id_entry.get())
    student_name = student_name_entry.get()
    student_age = student_age_spinbox.get()

    selected_subjects = [subject_name[i] for i, is_selected in enumerate(subject_selected) if is_selected.get()]

    # Begin the transaction
    try:
        mydb.start_transaction()

        # Inserting data into a table for each selected subject
        for subject in selected_subjects:
            sql = "INSERT INTO subject_registration (student_id, student_name, student_age, subject_name) VALUES (%s, %s, %s, %s)"
            val = (student_id, student_name, student_age, subject)
            cursor.execute(sql, val)
            print(f"Data inserted successfully for {subject}")

        # Commit the transaction after all data is inserted
        mydb.commit()
        print("All data inserted successfully!")
```



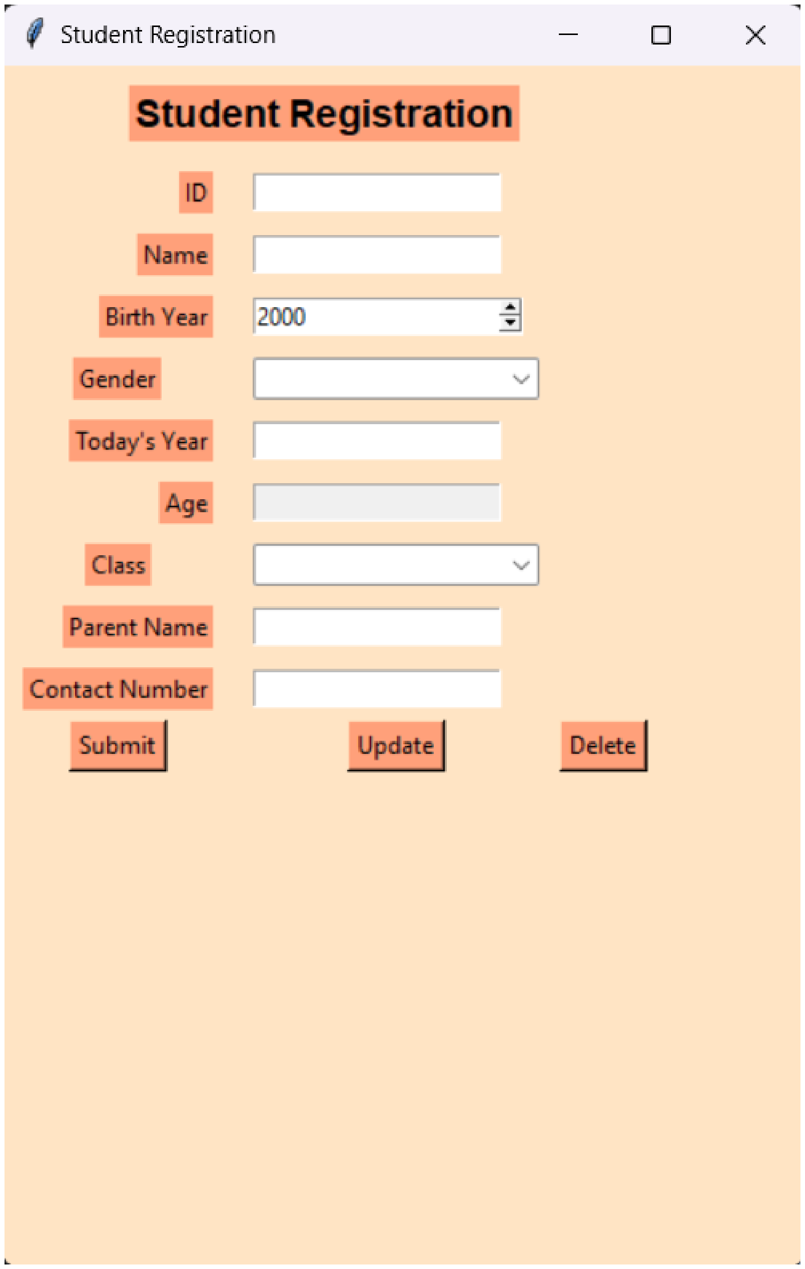
```python
    except mysql.connector.Error as err:
        print(f"Error: {err}")

        # Rollback the transaction in case of an error
        mydb.rollback()

    # Clear the entry fields after each insertion
    student_id_entry.delete(0, tk.END)
    student_name_entry.delete(0, tk.END)
    student_age_spinbox.delete(0, tk.END)
    for var in subject_selected:
        var.set(False)

def insert_data_in_loop():
    while True:
        subject_data()
        answer = messagebox.askquestion("Continue", "Do you want to insert data for another student?")
        if answer != 'yes':
            break

# Tkinter GUI
root = tk.Tk()
root.title("SUBJECT REGISTRATION")
root.geometry("400x600")
root.configure(bg='#C1FFC1')

# Page Title
label = tk.Label(root, text='SUBJECT REGISTRATION', font=("Sans", 14, "bold"), bg='#698B69')
label.pack(ipadx=10, ipady=15)

label_student_id = tk.Label(root, text="Student ID")
label_student_id.pack()
student_id_entry = tk.Entry(root)
student_id_entry.pack()

```



```python
label_student_name = tk.Label(root, text="Student Name")
label_student_name.pack()
student_name_entry = tk.Entry(root)
student_name_entry.pack()

# Student Age
age_label = tk.Label(root, text="Student Age :")
age_label.pack(pady=10)
student_age_spinbox = tk.Spinbox(root, from_=4, to=6)
student_age_spinbox.pack()

# Subject Name with Checkbuttons
label_subject = tk.Label(root, text="Select Subjects")
label_subject.pack(pady=10)

subject_name = [
    "ART",
    "SCIENCE",
    "READING",
    "WRITING",
    "MATHEMATICS",
]

subject_selected = [tk.BooleanVar() for _ in subject_name]

for i, subject in enumerate(subject_name):
    checkbox = tk.Checkbutton(root, text=subject, variable=subject_selected[i])
    checkbox.pack(pady=3)

# Button to insert data in a loop until the user decides to stop
insert_button_loop = tk.Button(root, text="Insert Data in Loop",  bg='#9BCD9B',command=insert_data_in_loop)
insert_button_loop.pack(pady=10)


root.mainloop()
```

6.0 SHAPSHOT OF GUI

6.1 Student Registration

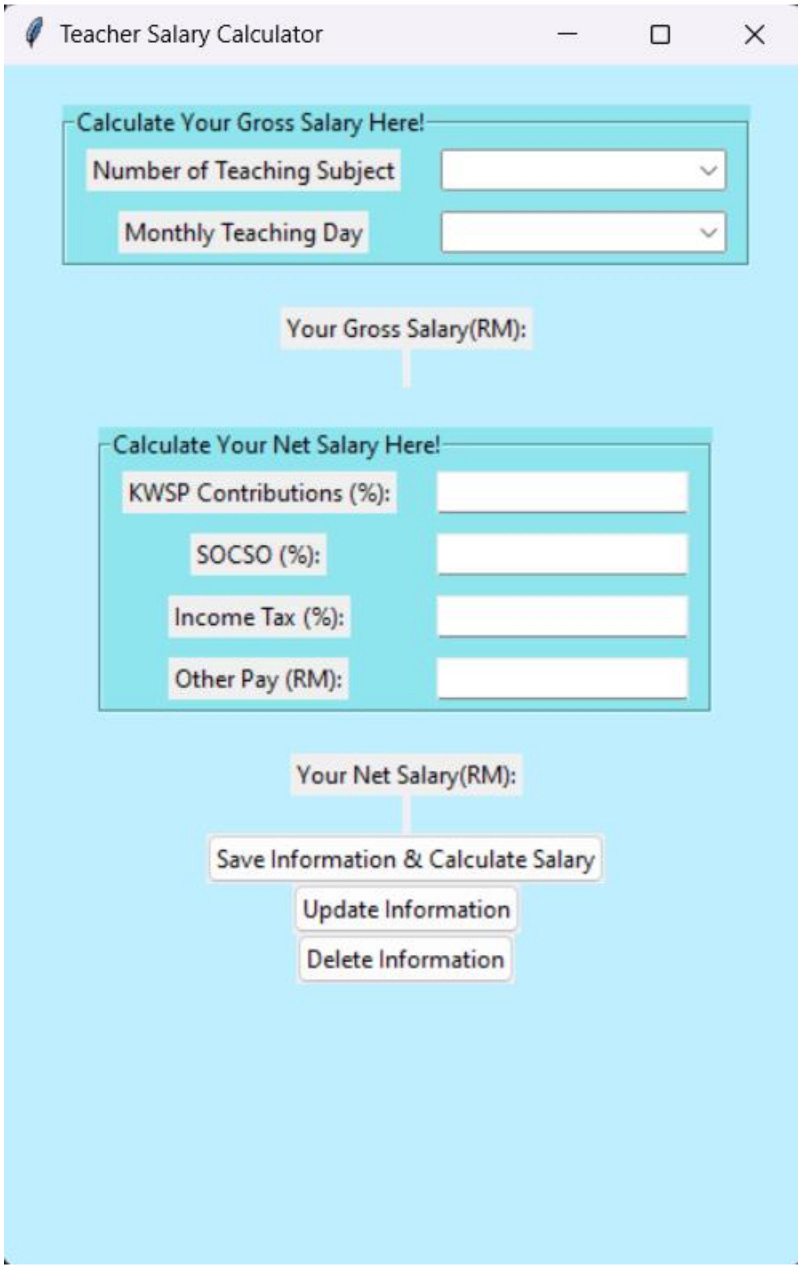


Student Registration

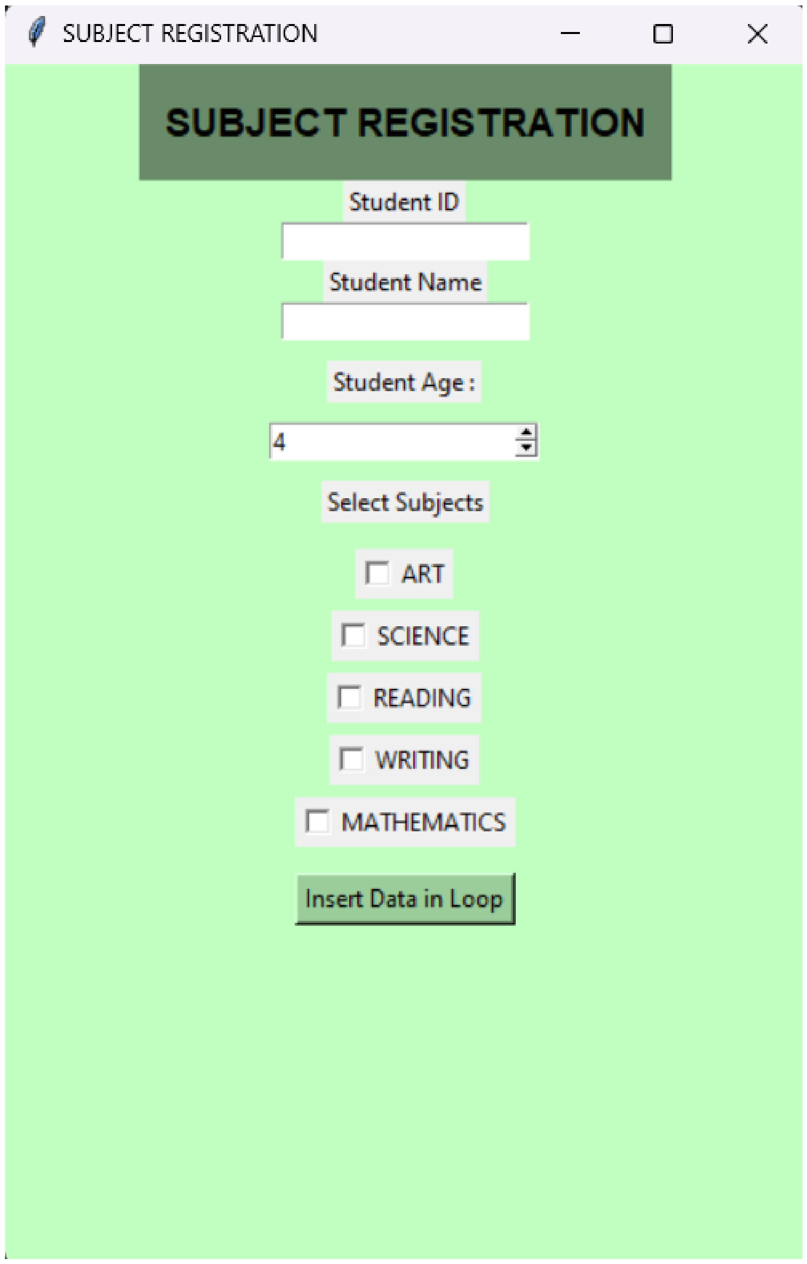
Student Registration

ID	
Name	
Birth Year	2000
Gender	
Today's Year	
Age	
Class	
Parent Name	
Contact Number	

6.2 Teacher Salary Calculator

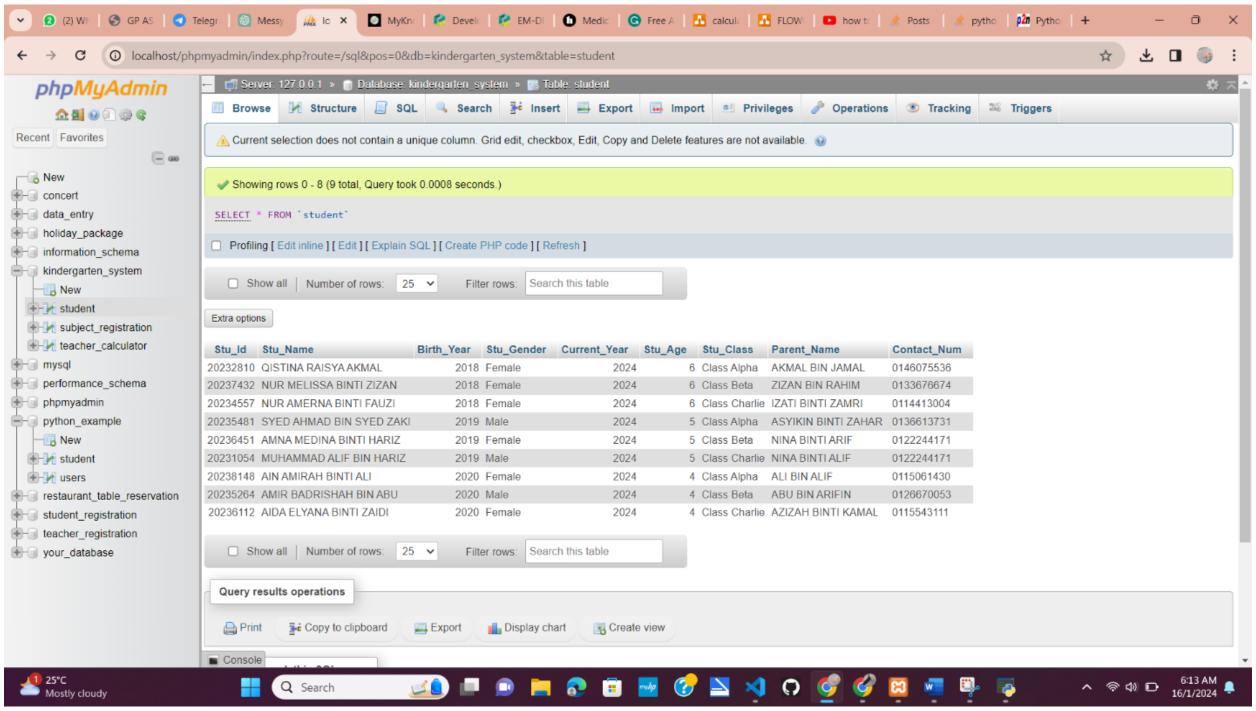


6.3 Subject Registration



7.0 SHAPSHOT OF DATABASE

7.1 Student Registration



localhost/phpmyadmin/index.php?route=/sql&pos=0&db=kindergarten_system&table=student

Server: 127.0.0.1 » Database: kindergarten_system » Table: student

Browse | Structure | SQL | Search | Insert | Export | Import | Privileges | Operations | Tracking | Triggers

Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

Showing rows 0 - 8 (9 total, Query took 0.0008 seconds.)

```
SELECT * FROM `student`
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table

Extra options

- New
- concert
- data_entry
- holiday_package
- information_schema
- kindergarten_system
  - New
  - student
  - subject_registration
  - teacher_calculator
- mysql
- performance_schema
- phpmyadmin
- python_example
  - New
  - student
  - users
- restaurant_table_reservation
- student_registration
- teacher_registration
- your_database

Stu_Id	Stu_Name	Birth_Year	Stu_Gender	Current_Year	Stu_Age	Stu_Class	Parent_Name	Contact_Num
20232810	QISTINA RAISYA AKMAL	2018	Female	2024	6	Class Alpha	AKMAL BIN JAMAL	0146075536
20237432	NUR MELISSA BINTI ZIZAN	2018	Female	2024	6	Class Beta	ZIZAN BIN RAHIM	0133676674
20234557	NUR AMERNA BINTI FAUZI	2018	Female	2024	6	Class Charlie	IZATI BINTI ZAMRI	0114413004
20235481	SYED AHMAD BIN SYED ZAKI	2019	Male	2024	5	Class Alpha	ASYIKIN BINTI ZAHAR	0136613731
20236451	AMNA MEDINA BINTI HARIZ	2019	Female	2024	5	Class Beta	NINA BINTI ARIF	0122244171
20231054	MUHAMMAD ALIF BIN HARIZ	2019	Male	2024	5	Class Charlie	NINA BINTI ALIF	0122244171
20238148	AIN AMIRAH BINTI ALI	2020	Female	2024	4	Class Alpha	ALI BIN ALIF	0115061430
20235264	AMIR BADRISHAH BIN ABU	2020	Male	2024	4	Class Beta	ABU BIN ARIFIN	0126670053
20236112	AIDA ELYANA BINTI ZAIDI	2020	Female	2024	4	Class Charlie	AZIZAH BINTI KAMAL	0115543111

Show all | Number of rows: 25 | Filter rows: Search this table

Query results operations

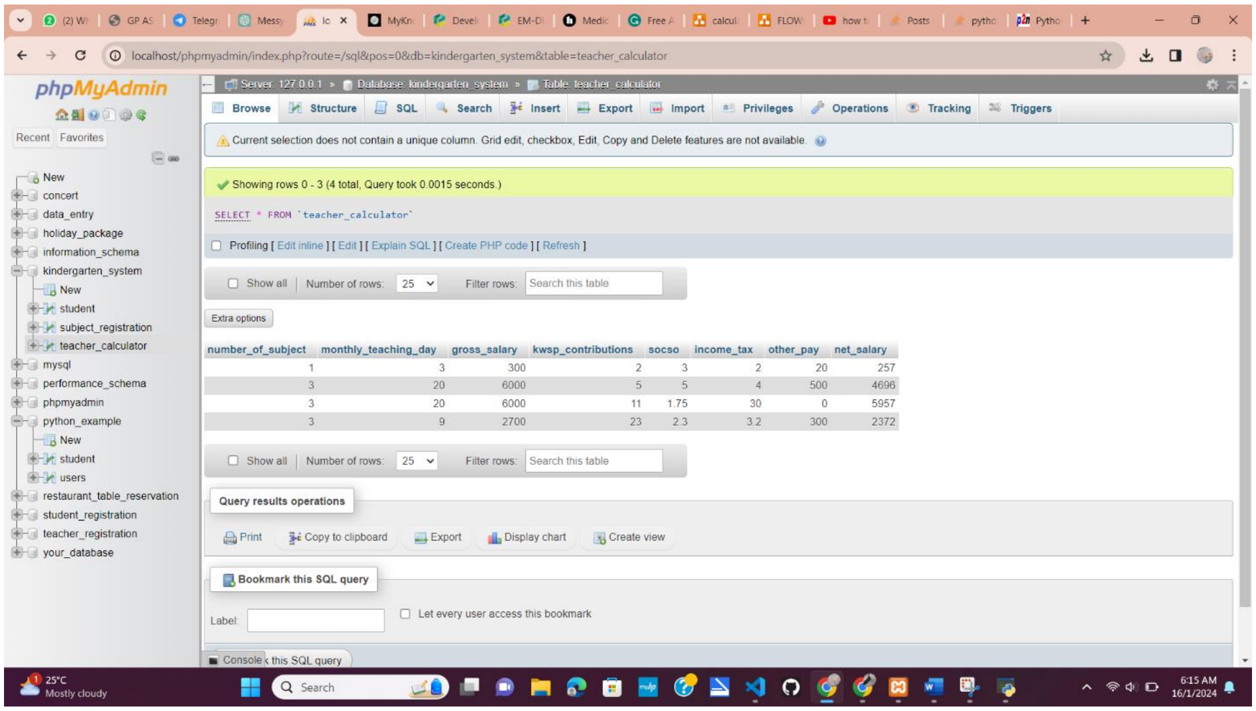
Print | Copy to clipboard | Export | Display chart | Create view

Console

25°C Mostly cloudy | Search | 6:13 AM 16/1/2024

Stu_Id	Stu_Name	Birth_Year	Stu_Gender	Current_Year	Stu_Age	Stu_Class	Parent_Name	Contact_Num
20232810	QISTINA RAISYA AKMAL	2018	Female	2024	6	Class Alpha	AKMAL BIN JAMAL	0146075536
20237432	NUR MELISSA BINTI ZIZAN	2018	Female	2024	6	Class Beta	ZIZAN BIN RAHIM	0133676674
20234557	NUR AMERNA BINTI FAUZI	2018	Female	2024	6	Class Charlie	IZATI BINTI ZAMRI	0114413004
20235481	SYED AHMAD BIN SYED ZAKI	2019	Male	2024	5	Class Alpha	ASYIKIN BINTI ZAHAR	0136613731
20236451	AMNA MEDINA BINTI HARIZ	2019	Female	2024	5	Class Beta	NINA BINTI ARIF	0122244171
20231054	MUHAMMAD ALIF BIN HARIZ	2019	Male	2024	5	Class Charlie	NINA BINTI ALIF	0122244171
20238148	AIN AMIRAH BINTI ALI	2020	Female	2024	4	Class Alpha	ALI BIN ALIF	0115061430
20235264	AMIR BADRISHAH BIN ABU	2020	Male	2024	4	Class Beta	ABU BIN ARIFIN	0126670053
20236112	AIDA ELYANA BINTI ZAIDI	2020	Female	2024	4	Class Charlie	AZIZAH BINTI KAMAL	0115543111

7.2 Teacher Salary Calculator



localhost/phpmyadmin/index.php?route=/sql&pos=0&db=kindergarten_system&table=teacher_calculator

Server: 127.0.0.1 » Database: kindergarten_system » Table: teacher_calculator

Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

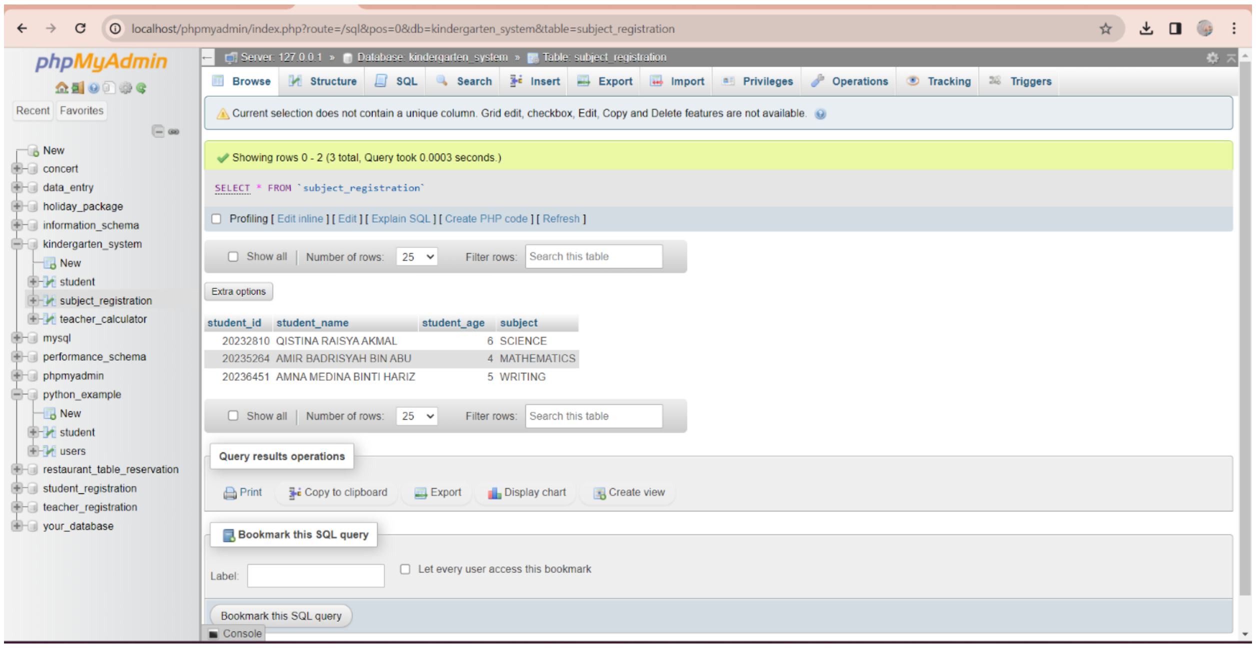
Showing rows 0 - 3 (4 total, Query took 0.0015 seconds.)

```
SELECT * FROM `teacher_calculator`
```

number_of_subject	monthly_teaching_day	gross_salary	kwsp_contributions	socso	income_tax	other_pay	net_salary
1	3	300	2	3	2	20	257
3	20	6000	5	5	4	500	4696
3	20	6000	11	1.75	30	0	5957
3	9	2700	23	2.3	3.2	300	2372

number_of_subject	monthly_teaching_day	gross_salary	kwsp_contributions	socso	income_tax	other_pay	net_salary
1	3	300	2	3	2	20	257
3	20	6000	5	5	4	500	4696
3	20	6000	11	1.75	30	0	5957
3	9	2700	23	2.3	3.2	300	2372

7.3 Subject Registration



localhost/phpmyadmin/index.php?route=/sql&pos=0&db=kindergarten_system&table=subject_registration

Server: 127.0.0.1 » Database: kindergarten_system » Table: subject_registration

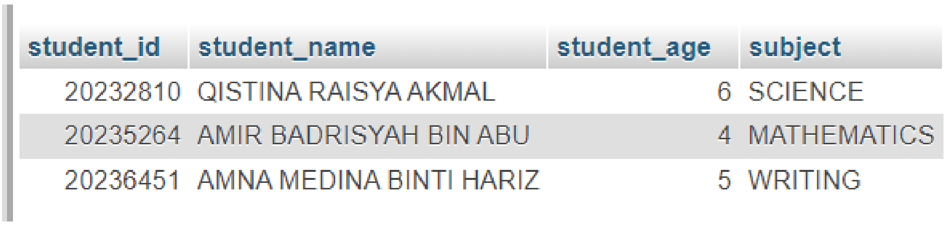
Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

Showing rows 0 - 2 (3 total, Query took 0.0003 seconds.)

```
SELECT * FROM `subject_registration`
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

student_id	student_name	student_age	subject
20232810	QISTINA RAISYA AKMAL	6	SCIENCE
20235264	AMIR BADRISYAH BIN ABU	4	MATHEMATICS
20236451	AMNA MEDINA BINTI HARIZ	5	WRITING



student_id	student_name	student_age	subject
20232810	QISTINA RAISYA AKMAL	6	SCIENCE
20235264	AMIR BADRISYAH BIN ABU	4	MATHEMATICS
20236451	AMNA MEDINA BINTI HARIZ	5	WRITING

8.0 CONCLUSION

In conclusion, this project has been a collaborative effort to address and improve key challenges within the kindergarten environment. Through the integration of coding, flowchart design, and database implementation, our team has worked diligently to enhance the registration process, simplify salary calculations for teachers, and provide students with a more accessible platform for subject registration.

As we conclude this assignment, we reflect on the progress made and the impact our solutions may have on the kindergarten community. The systematic registration process ensures that no one is left behind, promoting inclusivity and efficiency. Teachers can now navigate their salary calculations with ease, allowing them to focus more on their invaluable role in shaping young minds. Students also benefit from a user-friendly platform that facilitates informed decisions regarding their subject choices.

As we look toward the future, we remain committed to the ongoing exploration and application of technology in educational settings. By leveraging the subject of programming for libraries, we can continue to innovate and create solutions that address the evolving needs of the kindergarten community. This project serves as a stepping stone towards a more efficient, accessible, and technologically empowered kindergarten experience.