UNIVERSITI TEKNOLOGI MARA

KEDAH BRANCH, CAMPUS SUNGAI PETANI

SCHOOL OF INFORMATION SCIENCE

COLLEGE OF COMPUTING, INFORMATICS AND MATHEMATICS

DIPLOMA IN INFORMATICS OF LIBRARY (CDIM144)

IML208: PROGRAMMING FOR LIBRARIES

GROUP PROJECT:
STUDENT 1 STOP CENTER

Prepared By:

NAME	STUDENT ID
NUR MAISARAH BINTI ABDUL MANAH	2022892948
NURUL ANIS ADEELA BINTI ROSELEE	2022606634
NOOR HASMURNI BINTI SYAKRI	2022449098
PRISCILLA ANN VINCENT	2022679716

GROUP KCDIM1443F

Prepared For:

SIR AIRUL SHAZWAN BIN NORSHAHIMI

Submission Date:

17 JANUARY 2024

STUDENT GRADE CALCULATION

PREPARED BY:

NAME	STUDENT ID
NUR MAISARAH BINTI ABDUL MANAH	2022892948
NURUL ANIS ADEELA BINTI ROSELEE	2022606634
NOOR HASMURNI	2022449098
PRISCILLA ANN VINCENT	2022679716

GROUP KCDIM1443F

IM144 – DIPLOMA IN INFORMATIVE LIBRARY

SCHOOL OF INFORMATION SCIENCE

COLLEGE OF COMPUTING, INFORMATICS AND MATHEMATICS

UNIVESITI TEKNOLOGI MARA (UITM)

KEDAH BRANCH, CAMPUS SUNGAI PETANI

ACKNOWLEDGEMENT

We would like to take this opportunity to express our gratitude to those who have supported and inspired us throughout the completion of this assignment.

First and foremost, we would like to thank Sir Airul Shazwan Bin Norshahimi, our instructor in this subject, for his guidance, expertise, and continuing support, your help has been instrumental in shaping this work.

We are also indebted to our classmates and friends who offered encouragement and engaged in valuable discussions on the subject matter. Your input has greatly contributed to the depth and quality of this assignment.

Furthermore, we want to express our heartfelt thanks to our families for their unwavering encouragement and understanding during the countless hours spent on this project.

Finally, and most importantly, we want to extend our appreciation to Allah SAW for giving us the necessary motivation and strength to complete this assignment.

This assignment would not have been completed without the collective support and encouragement from the aforementioned individuals. Thank you for being a part of this journey.

TABLE OF CONTENTS

1.0 INTRODUCTION

Our project aims to simplify manual operations in the field of information management, specifically for university students. By creating a student information database, we can make tasks easier, reduce manpower, and ensure efficiency in the process. This database will also help institutions make better and well-informed decisions with the help of accurate and authentic data. The insert_and_display_course_info method calculates the total credit hours for selected courses by iterating over the list of courses and accumulating the credit hours for each course using the formula total_credit_hours = sum(course["credit_hour"] for course in courses).

1.1 PROBLEM STATEMENT

Our project aims to address the following issues:

- Student information: Allow users to input or add their information into the database, such as names, student IDs, programs, subjects, lecturer names, and semesters.
- Grade calculation: Implement a grading system that calculates the total credit hours for each course.
- User-friendly interface: To facilitate users in easily interacting with the system, both command line and graphical user interfaces will be provided based on user preference.

1.2 OBJECTIVE

Our project objectives are to:

- Ensure data validation and error handling to provide accurate and error-free input while giving appropriate feedback to users.
- Provide good scalability that can handle a reasonable number of students' information and calculate the total credit hour even when its performance is increasing.
- Provide efficient data management for student information that enables recording and storing scores based on how many courses.

2.0 FLOWCHART

2.1 FLOWCHART FOR TABLE STUDENT

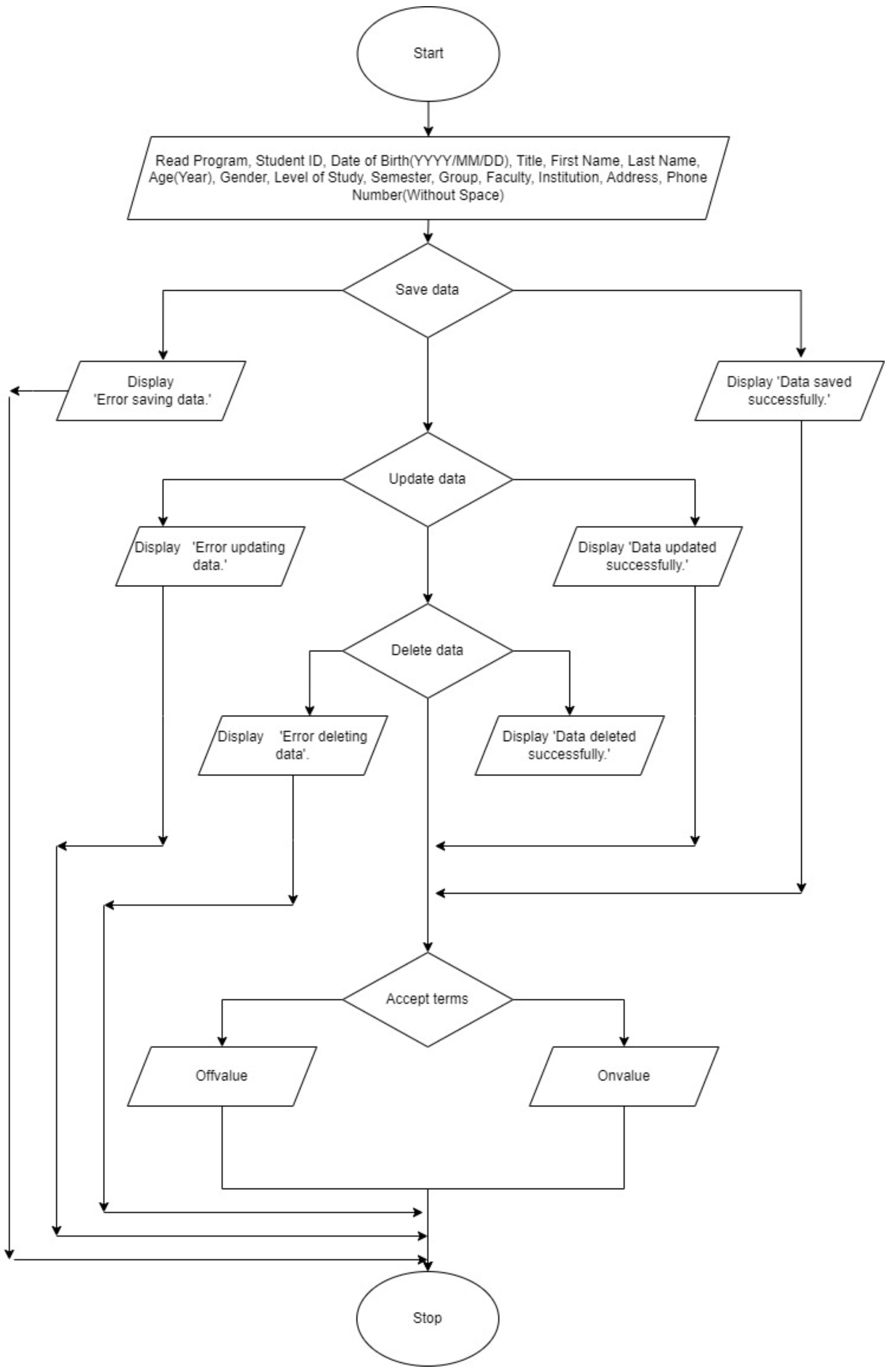

Start
Read Program, Student ID, Date of Birth(YYYY/MM/DD), Title, First Name, Last Name, Age(Year), Gender, Level of Study, Semester, Group, Faculty, Institution, Address, Phone Number(Without Space)
Save data
Display 'Error saving data.'
Display 'Data saved successfully.'
Update data
Display 'Error updating data.'
Display 'Data updated successfully.'
Delete data
Display 'Error deleting data'.
Display 'Data deleted successfully.'
Accept terms
Offvalue
Onvalue
Stop

Figure 2.1 Flowchart for table student

2.2 FLOWCHART FOR TABLE COURSE

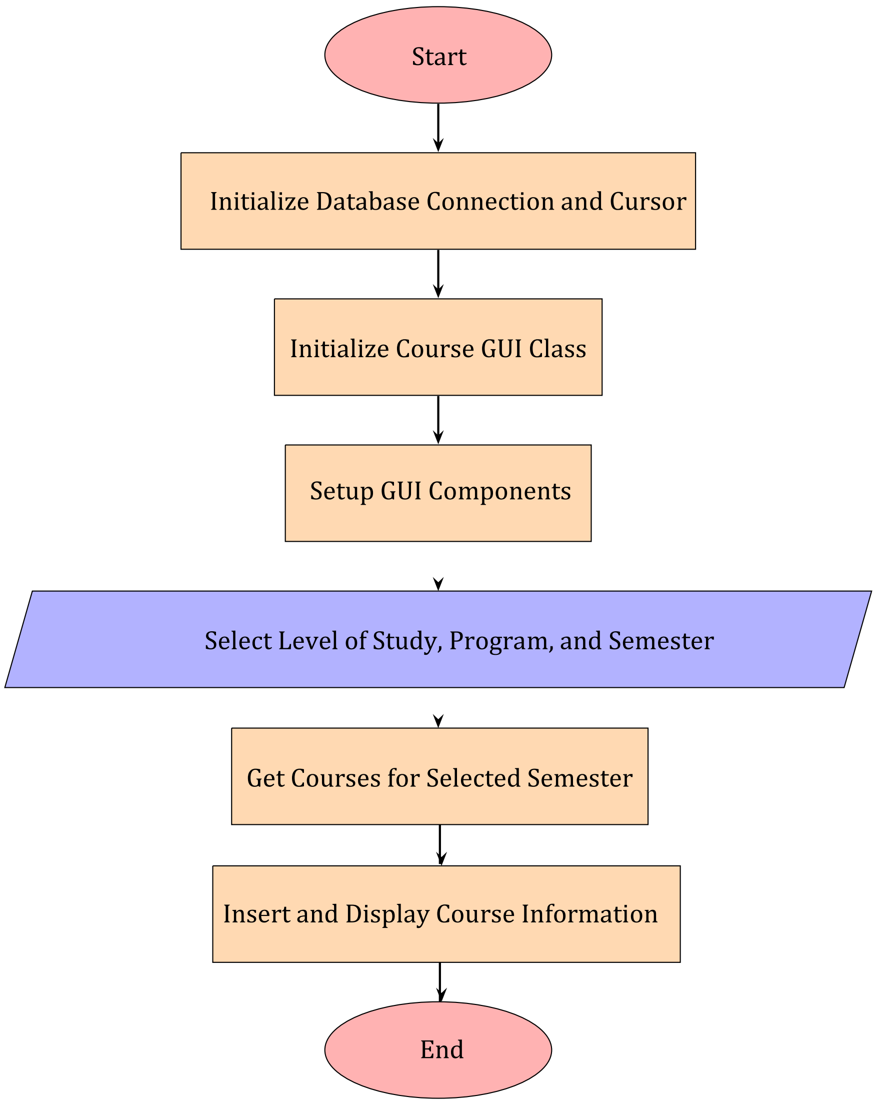

Start
Initialize Database Connection and Cursor
Initialize Course GUI Class
Setup GUI Components
Select Level of Study, Program, and Semester
Get Courses for Selected Semester
Insert and Display Course Information
End

Figure 1.2.1 Flowchart for table course

2.3 FLOWCHART FOR TABLE GRADE

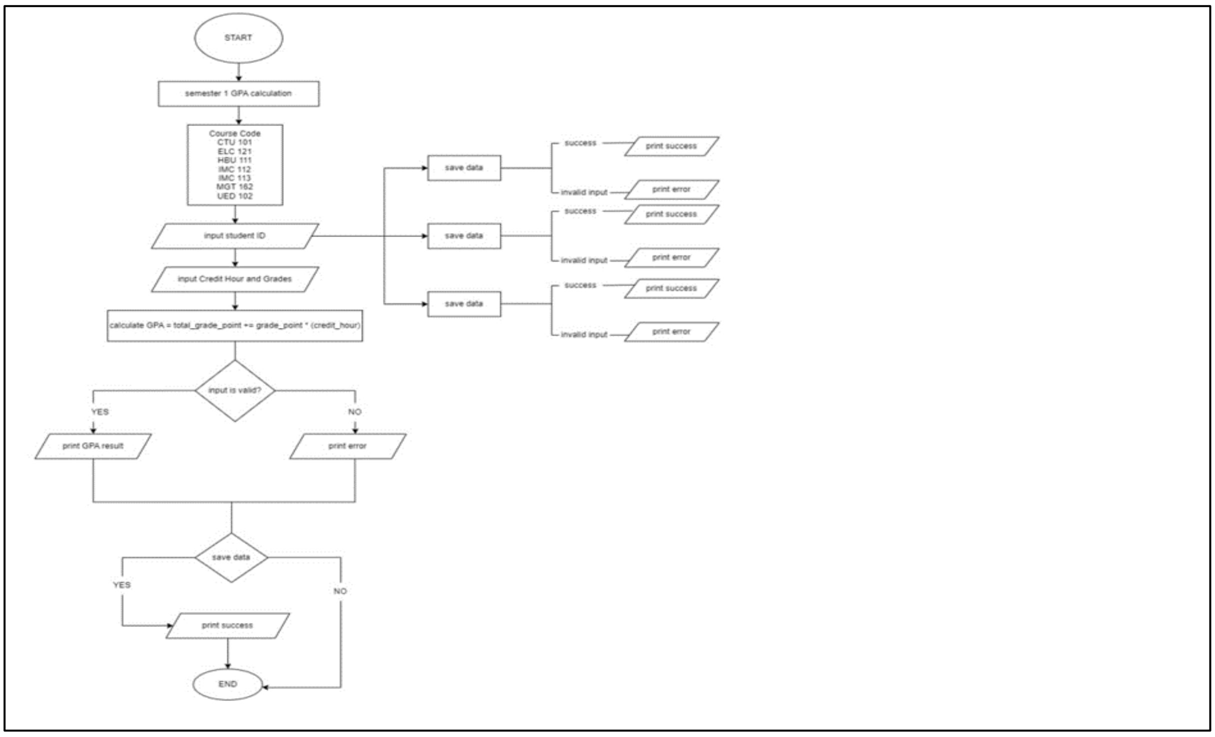


Figure 2.3.1 Flowchart for table grade_semester1

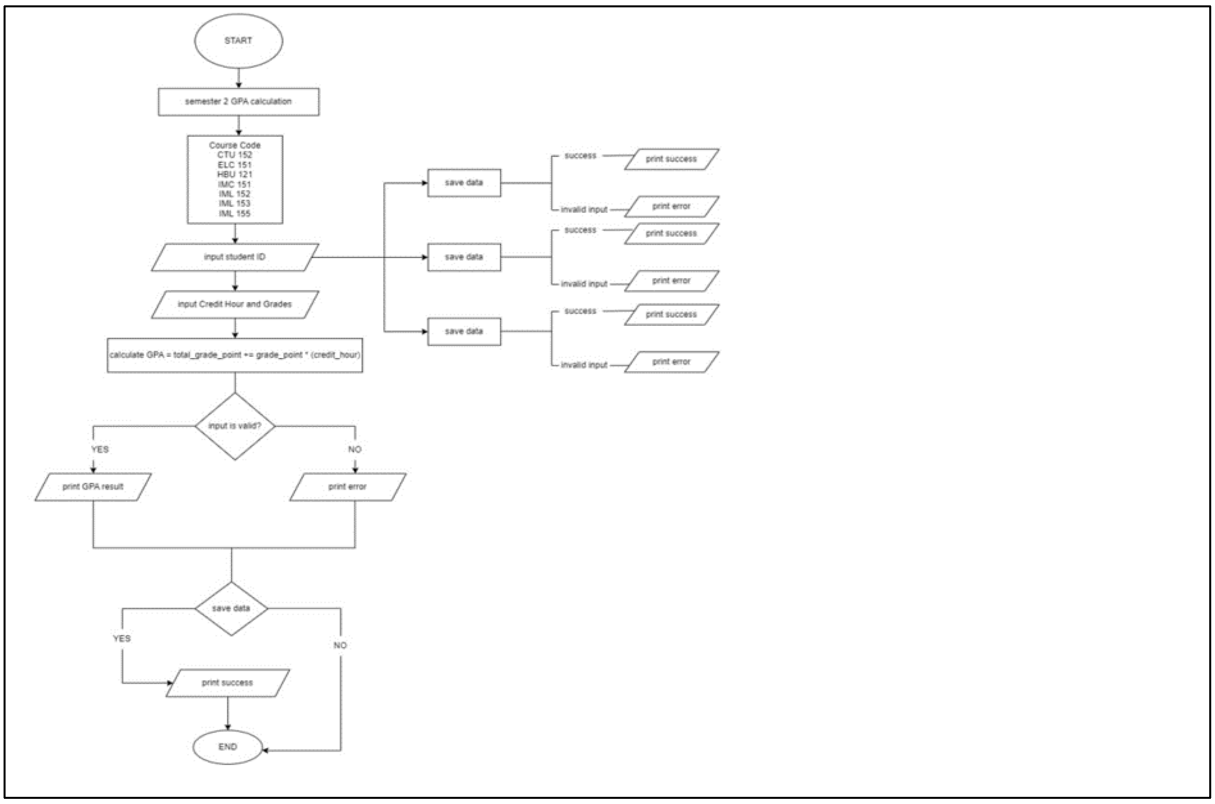


Figure 2.3.2 Flowchart for table grade_semester2

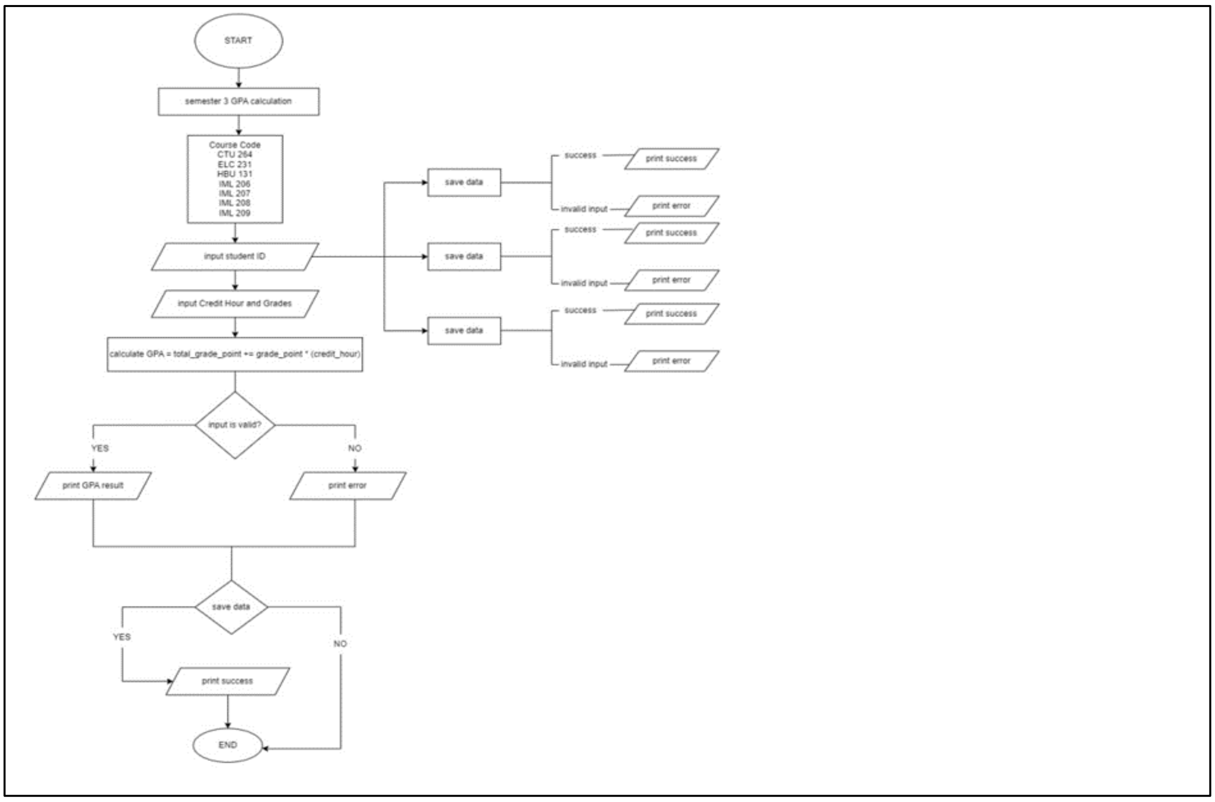


Figure 2.3.3 Flowchart for table grade_semester3

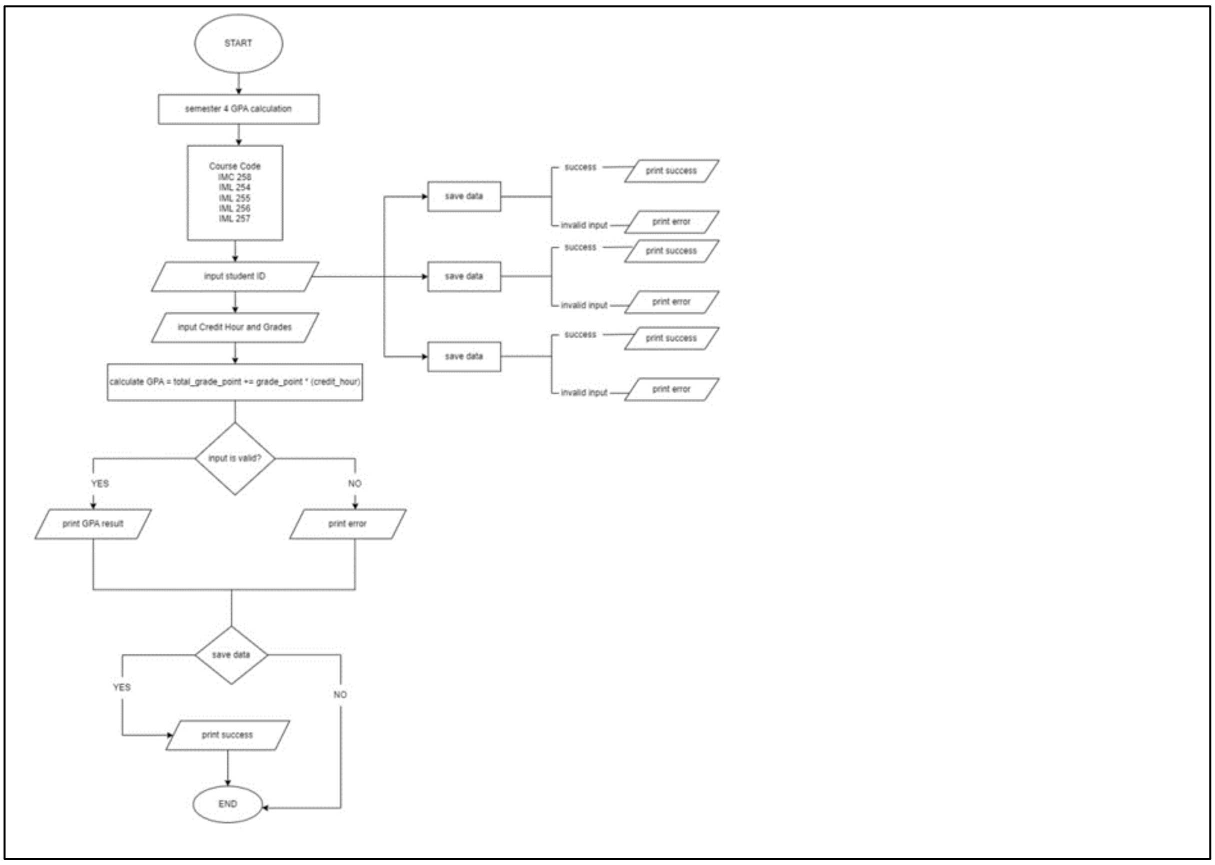


Figure 2.3.4 Flowchart for table grade_semester4

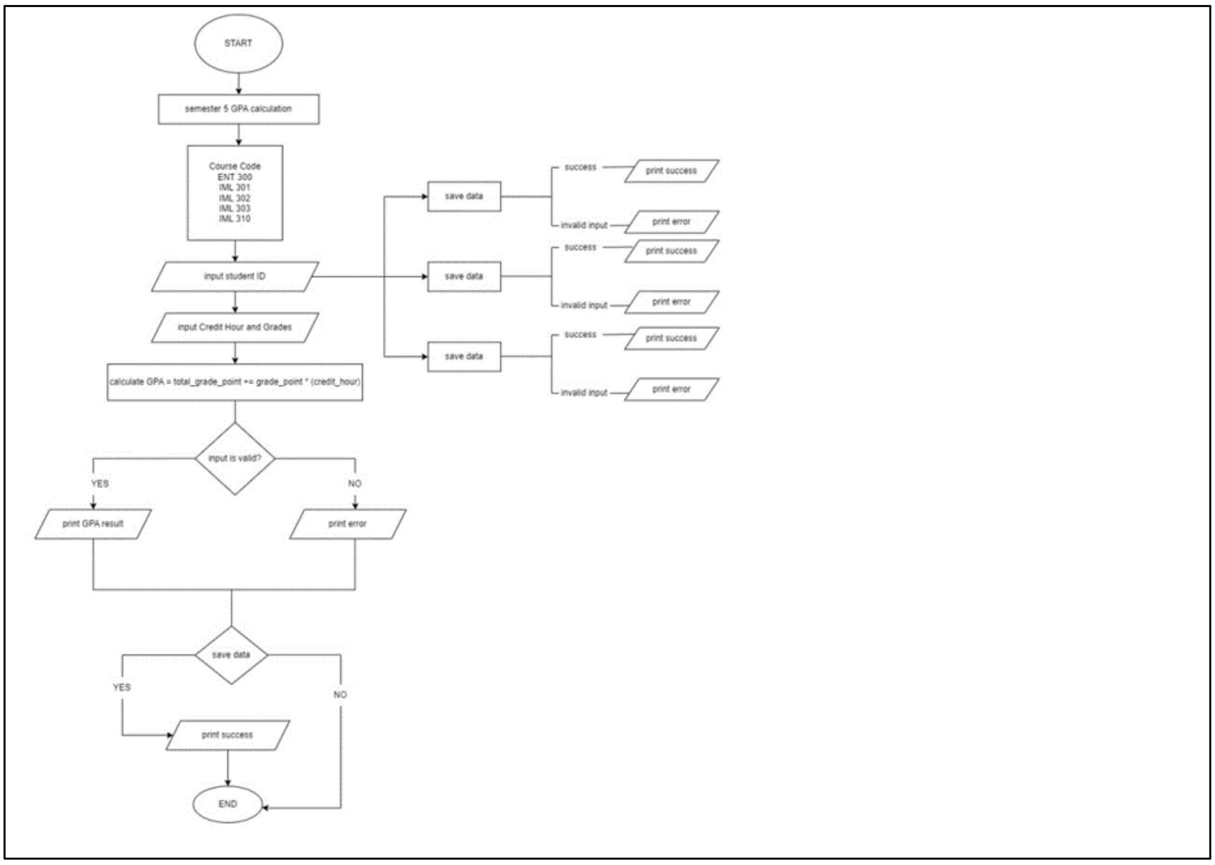


Figure 2.3.5 Flowchart for table grade_semester5

2.4 FLOWCHART FOR LISTS

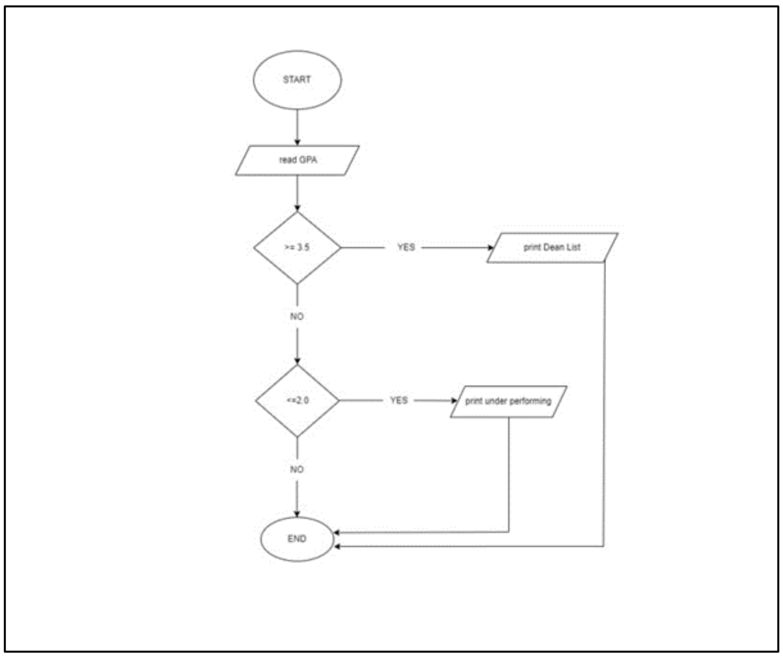
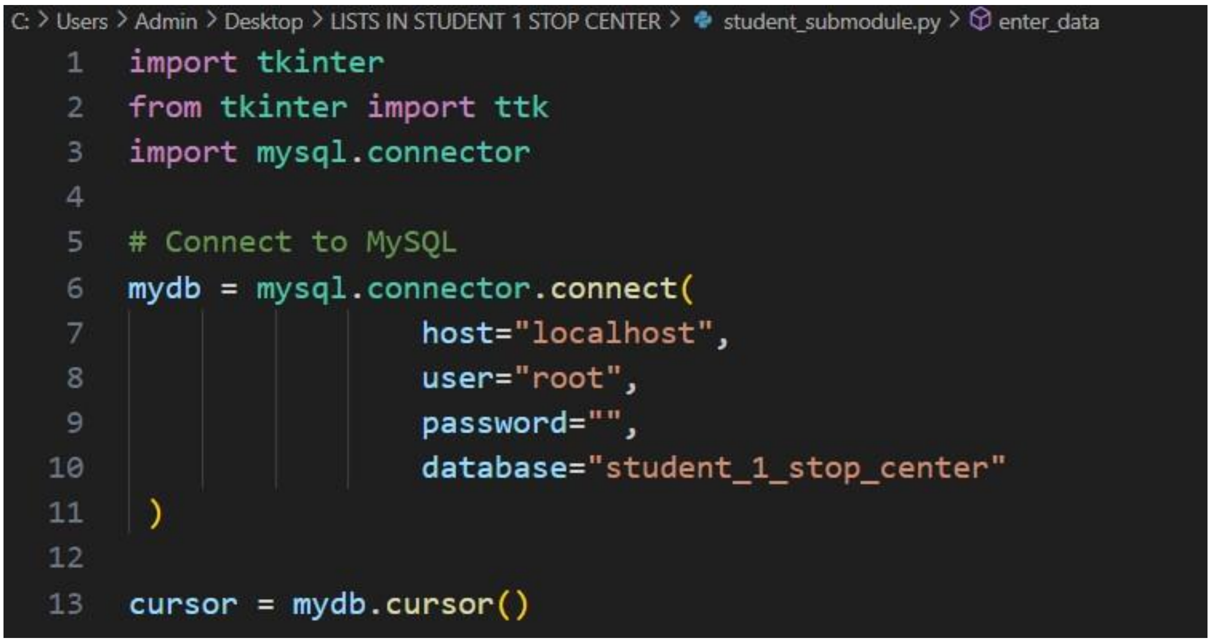


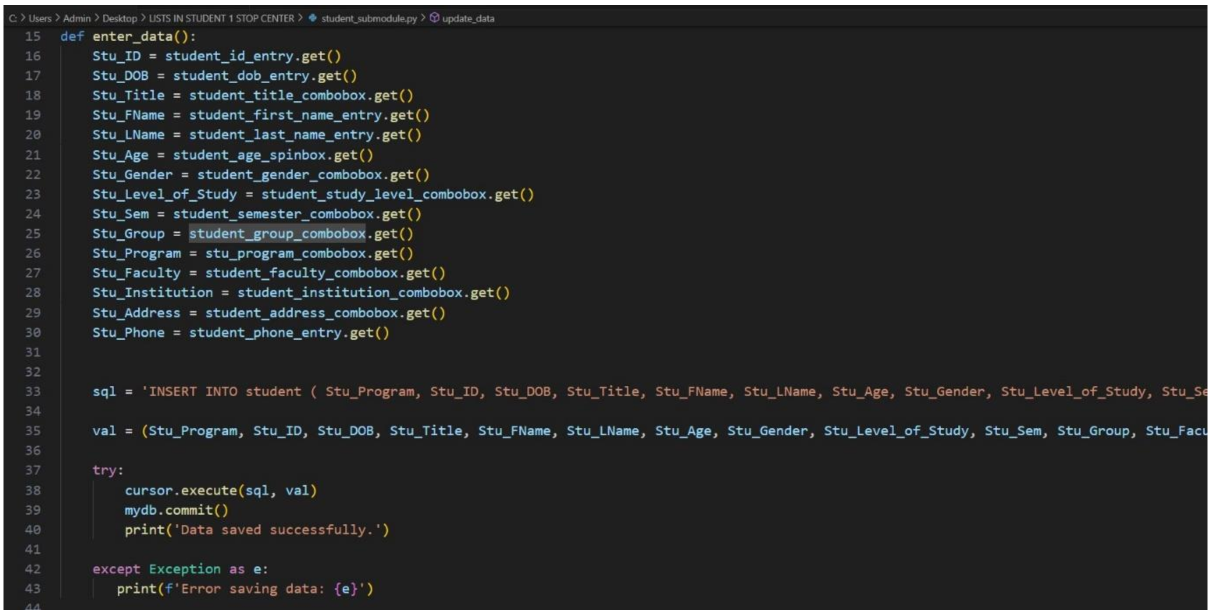
Figure 2.4.1 Flowchart for dean list and underperforming students's list

3.0 SNAPSHOTS OF PYTHON CODE

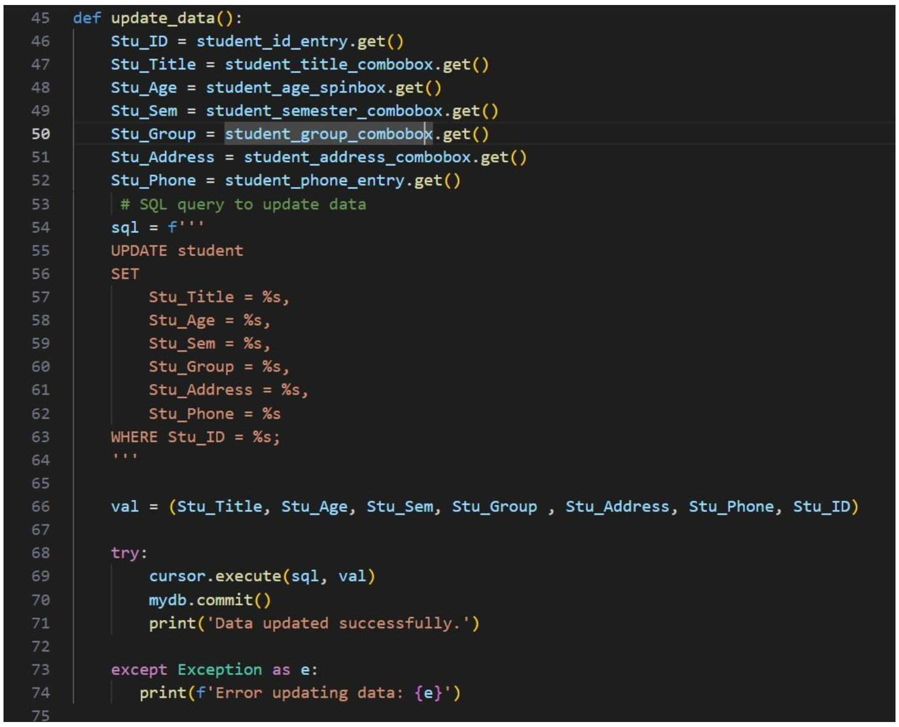
3.1 SNAPSHOTS OF PYTHON CODE FOR TABLE STUDENT


C: > Users > Admin > Desktop > LISTS IN STUDENT 1 STOP CENTER > student_submodule.py > enter_data

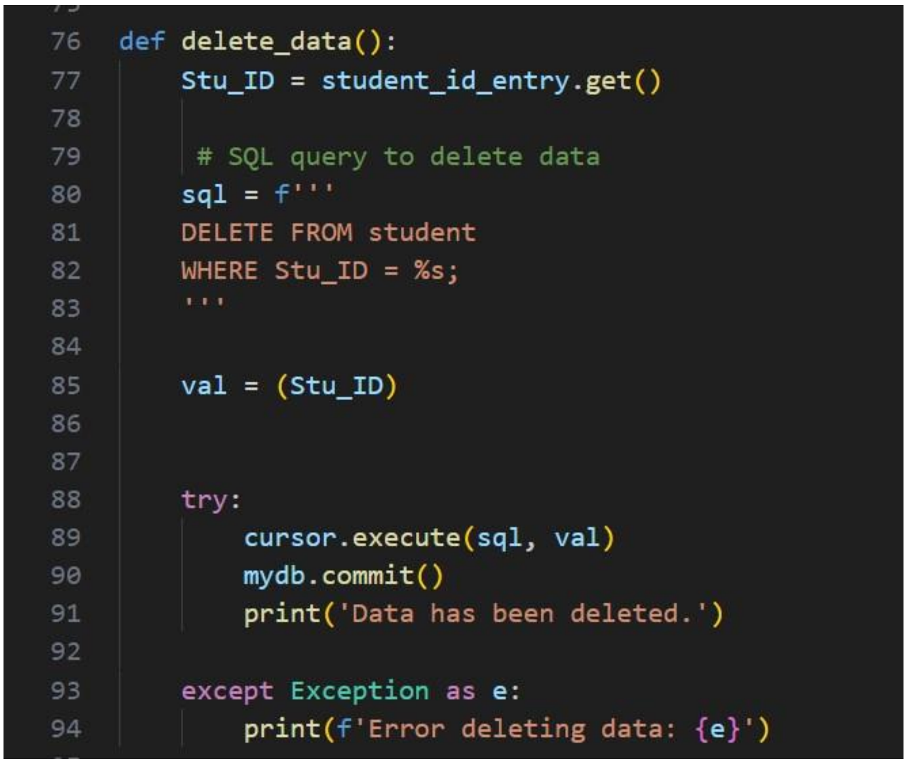
```python
import tkinter
from tkinter import ttk
import mysql.connector

# Connect to MySQL
mydb = mysql.connector.connect(
                host="localhost",
                user="root",
                password="",
                database="student_1_stop_center"
 )

cursor = mydb.cursor()
```


C: > Users > Admin > Desktop > LISTS IN STUDENT 1 STOP CENTER > student_submodule.py > update_data

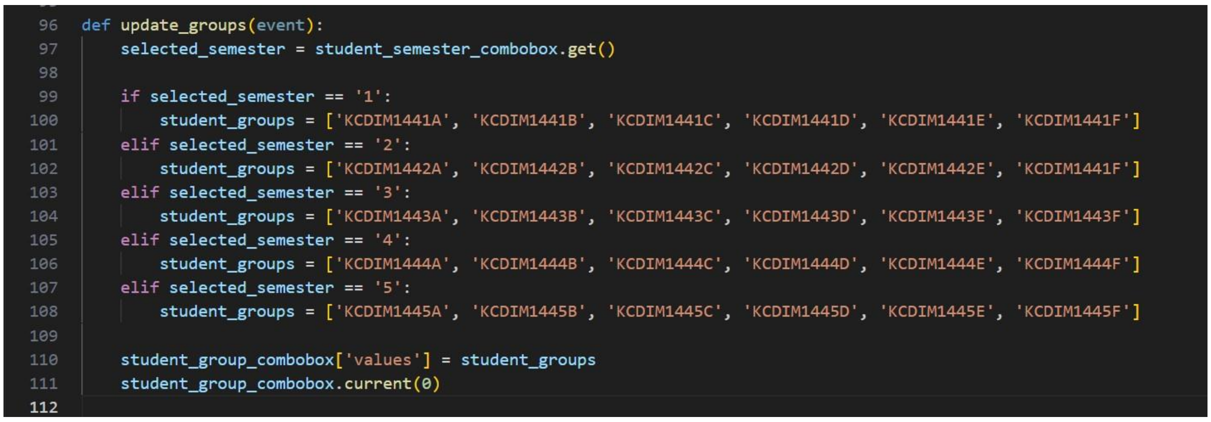
```python
def enter_data():
    Stu_ID = student_id_entry.get()
    Stu_DOB = student_dob_entry.get()
    Stu_Title = student_title_combobox.get()
    Stu_FName = student_first_name_entry.get()
    Stu_LName = student_last_name_entry.get()
    Stu_Age = student_age_spinbox.get()
    Stu_Gender = student_gender_combobox.get()
    Stu_Level_of_Study = student_study_level_combobox.get()
    Stu_Sem = student_semester_combobox.get()
    Stu_Group = student_group_combobox.get()
    Stu_Program = stu_program_combobox.get()
    Stu_Faculty = student_faculty_combobox.get()
    Stu_Institution = student_institution_combobox.get()
    Stu_Address = student_address_combobox.get()
    Stu_Phone = student_phone_entry.get()


    sql = 'INSERT INTO student ( Stu_Program, Stu_ID, Stu_DOB, Stu_Title, Stu_FName, Stu_LName, Stu_Age, Stu_Gender, Stu_Level_of_Study, Stu_Se

    val = (Stu_Program, Stu_ID, Stu_DOB, Stu_Title, Stu_FName, Stu_LName, Stu_Age, Stu_Gender, Stu_Level_of_Study, Stu_Sem, Stu_Group, Stu_Facu

    try:
        cursor.execute(sql, val)
        mydb.commit()
        print('Data saved successfully.')

    except Exception as e:
       print(f'Error saving data: {e}')
```



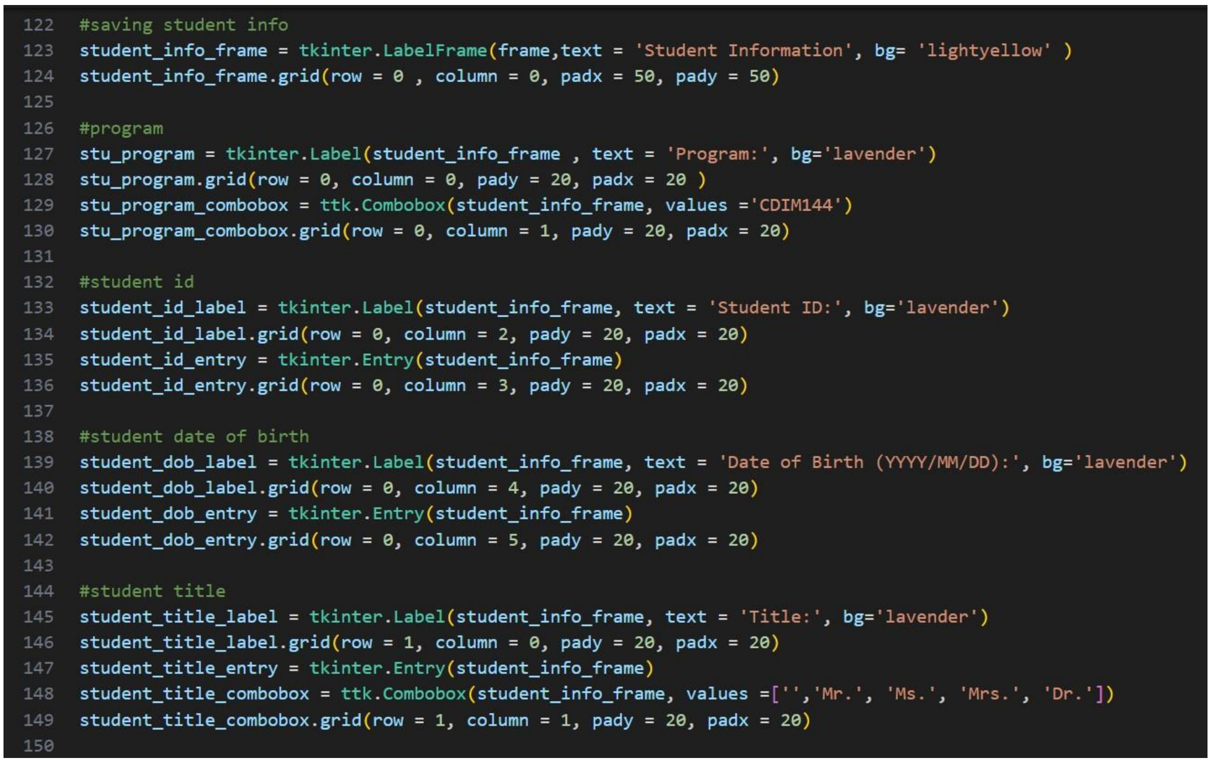
```python
def update_data():
    Stu_ID = student_id_entry.get()
    Stu_Title = student_title_combobox.get()
    Stu_Age = student_age_spinbox.get()
    Stu_Sem = student_semester_combobox.get()
    Stu_Group = student_group_combobox.get()
    Stu_Address = student_address_combobox.get()
    Stu_Phone = student_phone_entry.get()
     # SQL query to update data
    sql = f'''
    UPDATE student
    SET
        Stu_Title = %s,
        Stu_Age = %s,
        Stu_Sem = %s,
        Stu_Group = %s,
        Stu_Address = %s,
        Stu_Phone = %s
    WHERE Stu_ID = %s;
    '''

    val = (Stu_Title, Stu_Age, Stu_Sem, Stu_Group , Stu_Address, Stu_Phone, Stu_ID)

    try:
        cursor.execute(sql, val)
        mydb.commit()
        print('Data updated successfully.')

    except Exception as e:
       print(f'Error updating data: {e}')
```



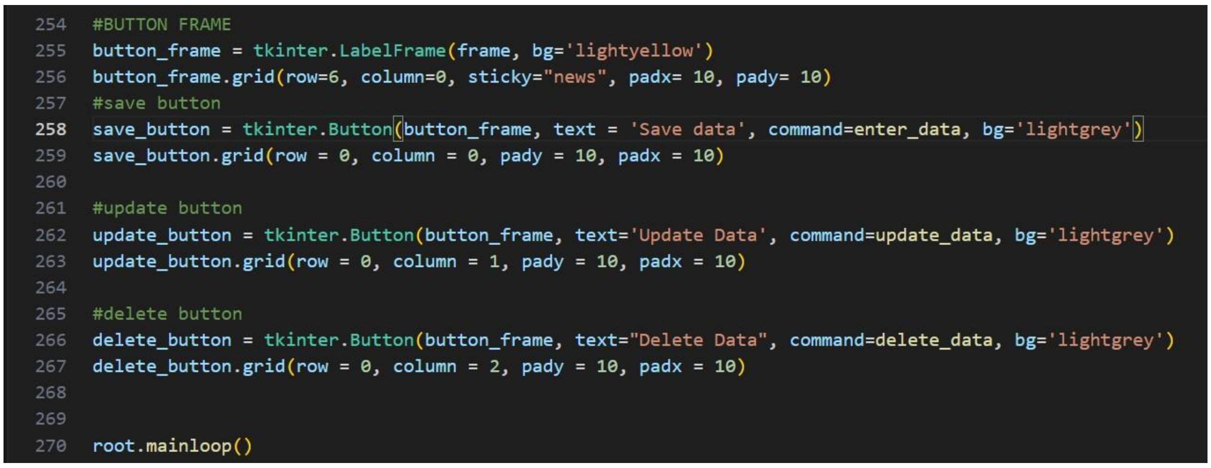
```python
def delete_data():
    Stu_ID = student_id_entry.get()

     # SQL query to delete data
    sql = f'''
    DELETE FROM student
    WHERE Stu_ID = %s;
    '''

    val = (Stu_ID)


    try:
        cursor.execute(sql, val)
        mydb.commit()
        print('Data has been deleted.')

    except Exception as e:
        print(f'Error deleting data: {e}')
```



```python
def update_groups(event):
    selected_semester = student_semester_combobox.get()

    if selected_semester == '1':
        student_groups = ['KCDIM1441A', 'KCDIM1441B', 'KCDIM1441C', 'KCDIM1441D', 'KCDIM1441E', 'KCDIM1441F']
    elif selected_semester == '2':
        student_groups = ['KCDIM1442A', 'KCDIM1442B', 'KCDIM1442C', 'KCDIM1442D', 'KCDIM1442E', 'KCDIM1441F']
    elif selected_semester == '3':
        student_groups = ['KCDIM1443A', 'KCDIM1443B', 'KCDIM1443C', 'KCDIM1443D', 'KCDIM1443E', 'KCDIM1443F']
    elif selected_semester == '4':
        student_groups = ['KCDIM1444A', 'KCDIM1444B', 'KCDIM1444C', 'KCDIM1444D', 'KCDIM1444E', 'KCDIM1444F']
    elif selected_semester == '5':
        student_groups = ['KCDIM1445A', 'KCDIM1445B', 'KCDIM1445C', 'KCDIM1445D', 'KCDIM1445E', 'KCDIM1445F']

    student_group_combobox['values'] = student_groups
    student_group_combobox.current(0)

```

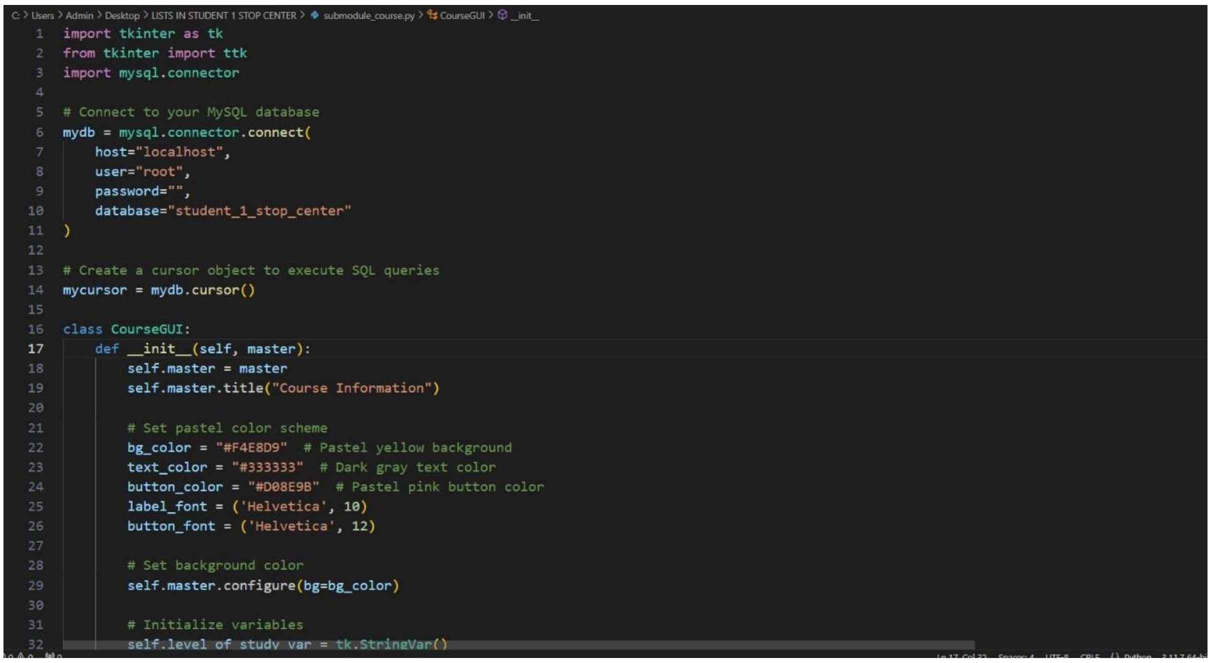


```python
#saving student info
student_info_frame = tkinter.LabelFrame(frame,text = 'Student Information', bg= 'lightyellow' )
student_info_frame.grid(row = 0 , column = 0, padx = 50, pady = 50)

#program
stu_program = tkinter.Label(student_info_frame , text = 'Program:', bg='lavender')
stu_program.grid(row = 0, column = 0, pady = 20, padx = 20 )
stu_program_combobox = ttk.Combobox(student_info_frame, values ='CDIM144')
stu_program_combobox.grid(row = 0, column = 1, pady = 20, padx = 20)

#student id
student_id_label = tkinter.Label(student_info_frame, text = 'Student ID:', bg='lavender')
student_id_label.grid(row = 0, column = 2, pady = 20, padx = 20)
student_id_entry = tkinter.Entry(student_info_frame)
student_id_entry.grid(row = 0, column = 3, pady = 20, padx = 20)

#student date of birth
student_dob_label = tkinter.Label(student_info_frame, text = 'Date of Birth (YYYY/MM/DD):', bg='lavender')
student_dob_label.grid(row = 0, column = 4, pady = 20, padx = 20)
student_dob_entry = tkinter.Entry(student_info_frame)
student_dob_entry.grid(row = 0, column = 5, pady = 20, padx = 20)

#student title
student_title_label = tkinter.Label(student_info_frame, text = 'Title:', bg='lavender')
student_title_label.grid(row = 1, column = 0, pady = 20, padx = 20)
student_title_entry = tkinter.Entry(student_info_frame)
student_title_combobox = ttk.Combobox(student_info_frame, values =['','Mr.', 'Ms.', 'Mrs.', 'Dr.'])
student_title_combobox.grid(row = 1, column = 1, pady = 20, padx = 20)

```



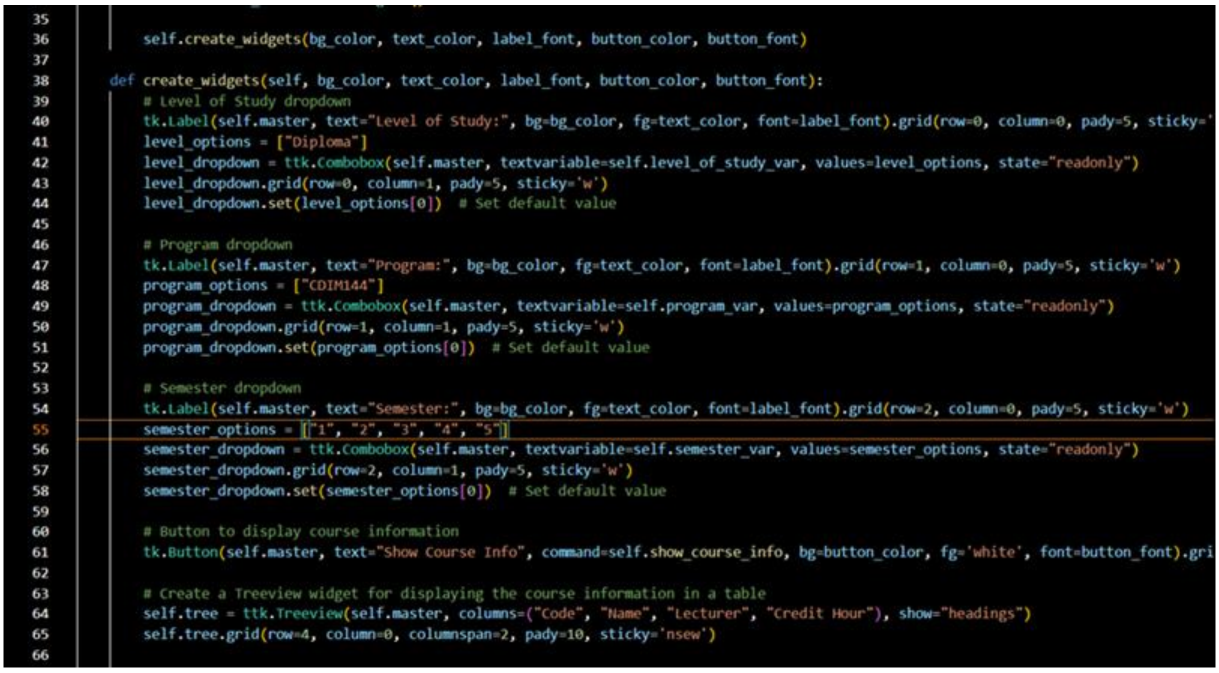
```python
#BUTTON FRAME
button_frame = tkinter.LabelFrame(frame, bg='lightyellow')
button_frame.grid(row=6, column=0, sticky="news", padx= 10, pady= 10)
#save button
save_button = tkinter.Button(button_frame, text = 'Save data', command=enter_data, bg='lightgrey')
save_button.grid(row = 0, column = 0, pady = 10, padx = 10)

#update button
update_button = tkinter.Button(button_frame, text='Update Data', command=update_data, bg='lightgrey')
update_button.grid(row = 0, column = 1, pady = 10, padx = 10)

#delete button
delete_button = tkinter.Button(button_frame, text="Delete Data", command=delete_data, bg='lightgrey')
delete_button.grid(row = 0, column = 2, pady = 10, padx = 10)


root.mainloop()
```

3.2 SNAPSHOTS OF PYTHON CODE FOR TABLE COURSE

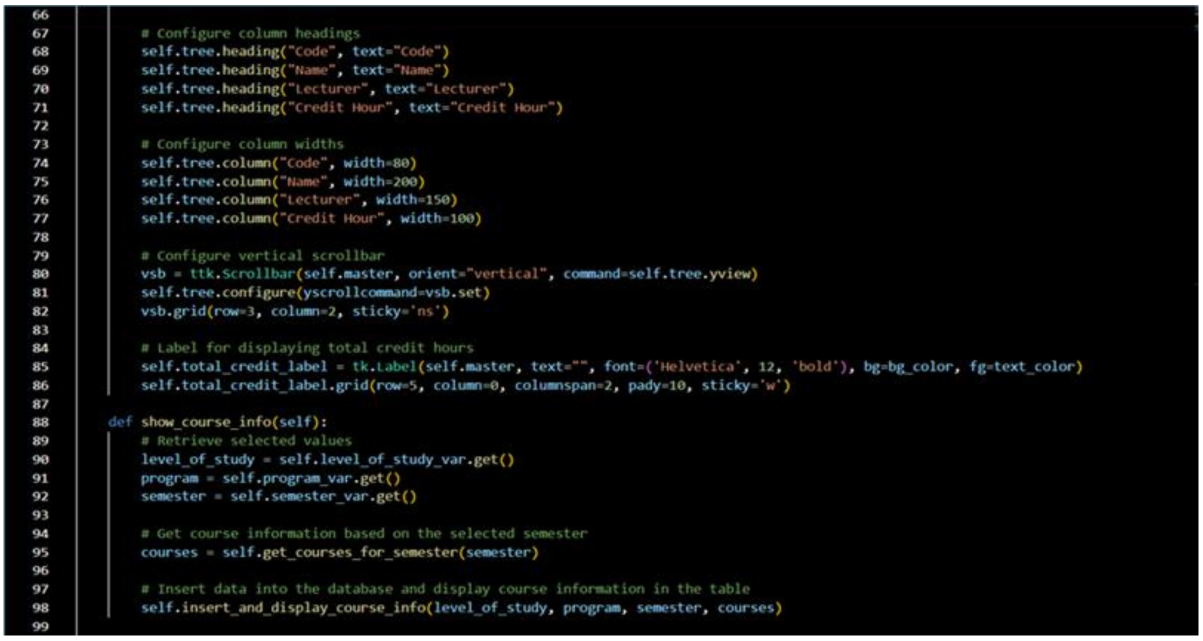


C: > Users > Admin > Desktop > LISTS IN STUDENT 1 STOP CENTER > submodule_course.py > CourseGUI > _init_

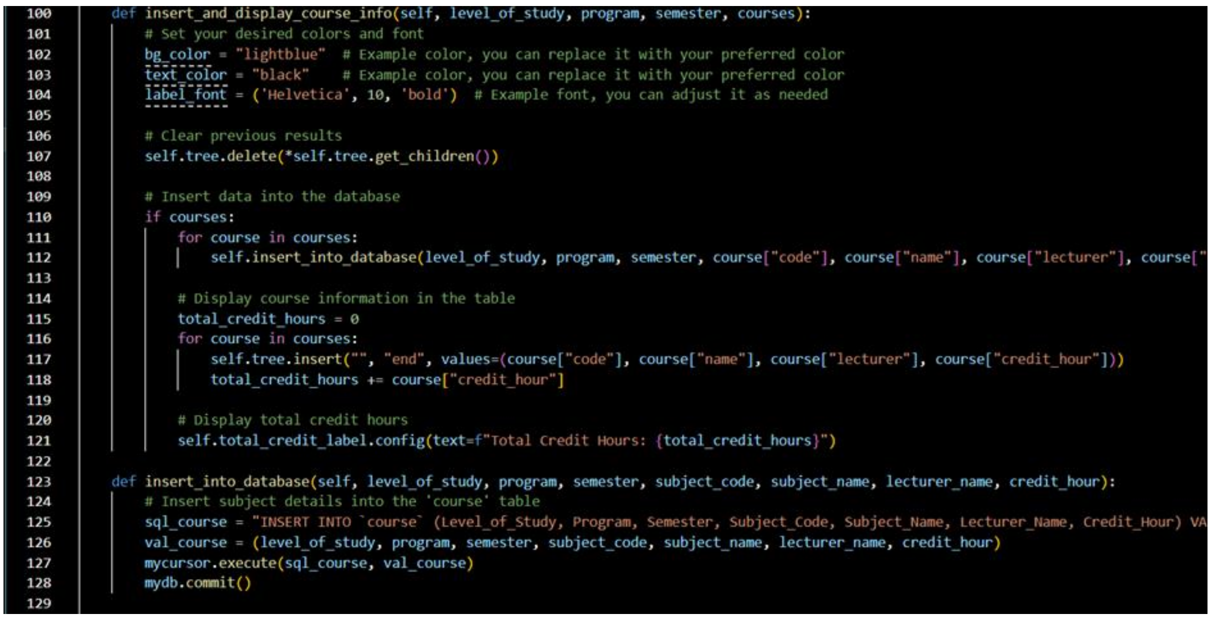
```python
import tkinter as tk
from tkinter import ttk
import mysql.connector

# Connect to your MySQL database
mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="student_1_stop_center"
)

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

class CourseGUI:
    def __init__(self, master):
        self.master = master
        self.master.title("Course Information")

        # Set pastel color scheme
        bg_color = "#F4E8D9"  # Pastel yellow background
        text_color = "#333333"  # Dark gray text color
        button_color = "#D08E9B"  # Pastel pink button color
        label_font = ('Helvetica', 10)
        button_font = ('Helvetica', 12)

        # Set background color
        self.master.configure(bg=bg_color)

        # Initialize variables
        self.level_of_study_var = tk.StringVar()
```



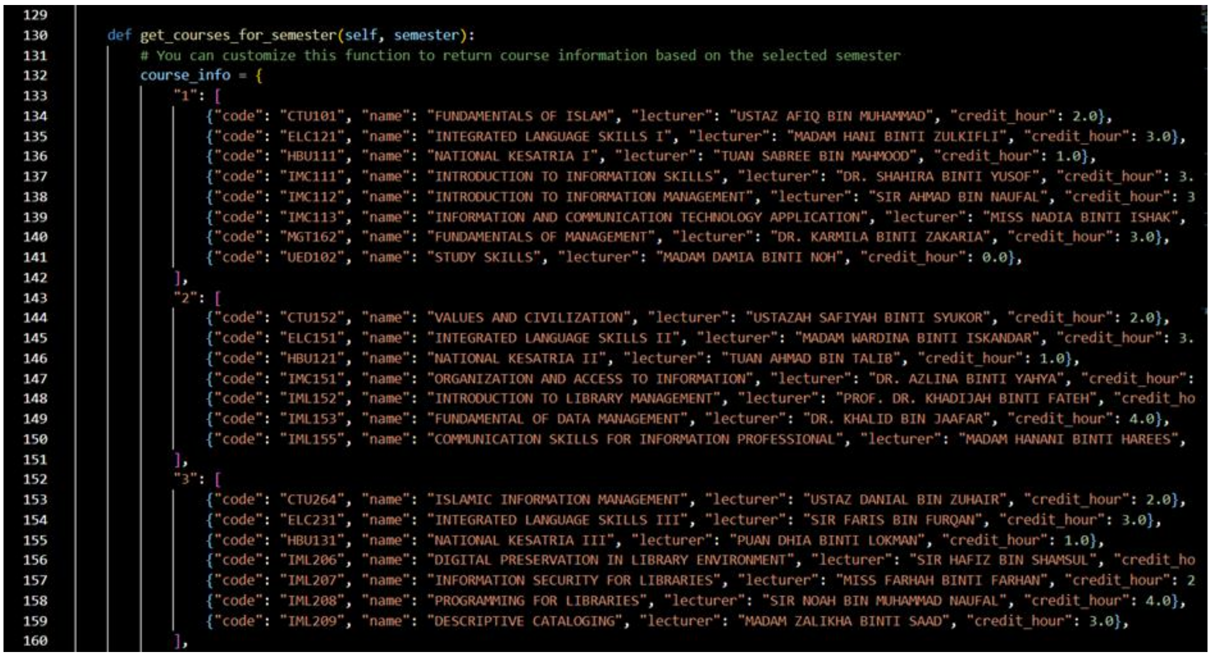
```python

        self.create_widgets(bg_color, text_color, label_font, button_color, button_font)

    def create_widgets(self, bg_color, text_color, label_font, button_color, button_font):
        # Level of Study dropdown
        tk.Label(self.master, text="Level of Study:", bg=bg_color, fg=text_color, font=label_font).grid(row=0, column=0, pady=5, sticky='
        level_options = ["Diploma"]
        level_dropdown = ttk.Combobox(self.master, textvariable=self.level_of_study_var, values=level_options, state="readonly")
        level_dropdown.grid(row=0, column=1, pady=5, sticky='w')
        level_dropdown.set(level_options[0])  # Set default value

        # Program dropdown
        tk.Label(self.master, text="Program:", bg=bg_color, fg=text_color, font=label_font).grid(row=1, column=0, pady=5, sticky='w')
        program_options = ["CDIM144"]
        program_dropdown = ttk.Combobox(self.master, textvariable=self.program_var, values=program_options, state="readonly")
        program_dropdown.grid(row=1, column=1, pady=5, sticky='w')
        program_dropdown.set(program_options[0])  # Set default value

        # Semester dropdown
        tk.Label(self.master, text="Semester:", bg=bg_color, fg=text_color, font=label_font).grid(row=2, column=0, pady=5, sticky='w')
        semester_options = ["1", "2", "3", "4", "5"]
        semester_dropdown = ttk.Combobox(self.master, textvariable=self.semester_var, values=semester_options, state="readonly")
        semester_dropdown.grid(row=2, column=1, pady=5, sticky='w')
        semester_dropdown.set(semester_options[0])  # Set default value

        # Button to display course information
        tk.Button(self.master, text="Show Course Info", command=self.show_course_info, bg=button_color, fg='white', font=button_font).gri

        # Create a Treeview widget for displaying the course information in a table
        self.tree = ttk.Treeview(self.master, columns=("Code", "Name", "Lecturer", "Credit Hour"), show="headings")
        self.tree.grid(row=4, column=0, columnspan=2, pady=10, sticky='nsew')

```



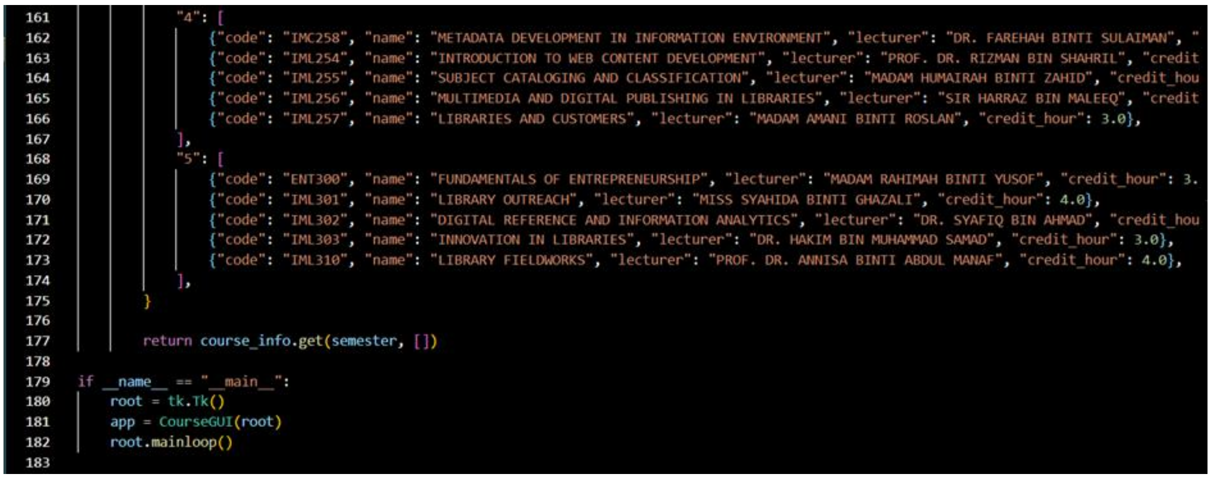
```python

        # Configure column headings
        self.tree.heading("Code", text="Code")
        self.tree.heading("Name", text="Name")
        self.tree.heading("Lecturer", text="Lecturer")
        self.tree.heading("Credit Hour", text="Credit Hour")

        # Configure column widths
        self.tree.column("Code", width=80)
        self.tree.column("Name", width=200)
        self.tree.column("Lecturer", width=150)
        self.tree.column("Credit Hour", width=100)

        # Configure vertical scrollbar
        vsb = ttk.Scrollbar(self.master, orient="vertical", command=self.tree.yview)
        self.tree.configure(yscrollcommand=vsb.set)
        vsb.grid(row=3, column=2, sticky='ns')

        # Label for displaying total credit hours
        self.total_credit_label = tk.Label(self.master, text="", font=('Helvetica', 12, 'bold'), bg=bg_color, fg=text_color)
        self.total_credit_label.grid(row=5, column=0, columnspan=2, pady=10, sticky='w')

    def show_course_info(self):
        # Retrieve selected values
        level_of_study = self.level_of_study_var.get()
        program = self.program_var.get()
        semester = self.semester_var.get()

        # Get course information based on the selected semester
        courses = self.get_courses_for_semester(semester)

        # Insert data into the database and display course information in the table
        self.insert_and_display_course_info(level_of_study, program, semester, courses)

```



```python
    def insert_and_display_course_info(self, level_of_study, program, semester, courses):
        # Set your desired colors and font
        bg_color = "lightblue"  # Example color, you can replace it with your preferred color
        text_color = "black"    # Example color, you can replace it with your preferred color
        label_font = ('Helvetica', 10, 'bold')  # Example font, you can adjust it as needed

        # Clear previous results
        self.tree.delete(*self.tree.get_children())

        # Insert data into the database
        if courses:
            for course in courses:
                self.insert_into_database(level_of_study, program, semester, course["code"], course["name"], course["lecturer"], course["

            # Display course information in the table
            total_credit_hours = 0
            for course in courses:
                self.tree.insert("", "end", values=(course["code"], course["name"], course["lecturer"], course["credit_hour"]))
                total_credit_hours += course["credit_hour"]

            # Display total credit hours
            self.total_credit_label.config(text=f"Total Credit Hours: {total_credit_hours}")

    def insert_into_database(self, level_of_study, program, semester, subject_code, subject_name, lecturer_name, credit_hour):
        # Insert subject details into the 'course' table
        sql_course = "INSERT INTO `course` (Level_of_Study, Program, Semester, Subject_Code, Subject_Name, Lecturer_Name, Credit_Hour) VA
        val_course = (level_of_study, program, semester, subject_code, subject_name, lecturer_name, credit_hour)
        mycursor.execute(sql_course, val_course)
        mydb.commit()

```

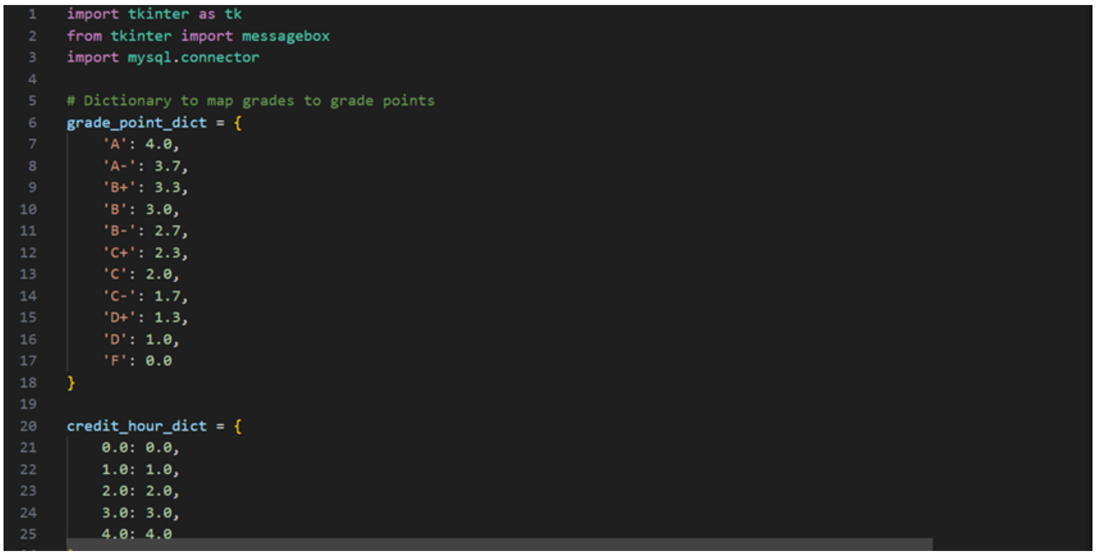


```

    def get_courses_for_semester(self, semester):
        # You can customize this function to return course information based on the selected semester
        course_info = {
            "1": [
                {"code": "CTU101", "name": "FUNDAMENTALS OF ISLAM", "lecturer": "USTAZ AFIQ BIN MUHAMMAD", "credit_hour": 2.0},
                {"code": "ELC121", "name": "INTEGRATED LANGUAGE SKILLS I", "lecturer": "MADAM HANI BINTI ZULKIFLI", "credit_hour": 3.0},
                {"code": "HBU111", "name": "NATIONAL KESATRIA I", "lecturer": "TUAN SABREE BIN MAHMOOD", "credit_hour": 1.0},
                {"code": "IMC111", "name": "INTRODUCTION TO INFORMATION SKILLS", "lecturer": "DR. SHAHIRA BINTI YUSOF", "credit_hour": 3.
                {"code": "IMC112", "name": "INTRODUCTION TO INFORMATION MANAGEMENT", "lecturer": "SIR AHMAD BIN NAUFAL", "credit_hour": 3
                {"code": "IMC113", "name": "INFORMATION AND COMMUNICATION TECHNOLOGY APPLICATION", "lecturer": "MISS NADIA BINTI ISHAK",
                {"code": "MGT162", "name": "FUNDAMENTALS OF MANAGEMENT", "lecturer": "DR. KARMILA BINTI ZAKARIA", "credit_hour": 3.0},
                {"code": "UED102", "name": "STUDY SKILLS", "lecturer": "MADAM DAMIA BINTI NOH", "credit_hour": 0.0},
            ],
            "2": [
                {"code": "CTU152", "name": "VALUES AND CIVILIZATION", "lecturer": "USTAZAH SAFIYAH BINTI SYUKOR", "credit_hour": 2.0},
                {"code": "ELC151", "name": "INTEGRATED LANGUAGE SKILLS II", "lecturer": "MADAM WARDINA BINTI ISKANDAR", "credit_hour": 3.
                {"code": "HBU121", "name": "NATIONAL KESATRIA II", "lecturer": "TUAN AHMAD BIN TALIB", "credit_hour": 1.0},
                {"code": "IMC151", "name": "ORGANIZATION AND ACCESS TO INFORMATION", "lecturer": "DR. AZLINA BINTI YAHYA", "credit_hour":
                {"code": "IML152", "name": "INTRODUCTION TO LIBRARY MANAGEMENT", "lecturer": "PROF. DR. KHADIJAH BINTI FATEH", "credit_ho
                {"code": "IML153", "name": "FUNDAMENTAL OF DATA MANAGEMENT", "lecturer": "DR. KHALID BIN JAAFAR", "credit_hour": 4.0},
                {"code": "IML155", "name": "COMMUNICATION SKILLS FOR INFORMATION PROFESSIONAL", "lecturer": "MADAM HANANI BINTI HAREES",
            ],
            "3": [
                {"code": "CTU264", "name": "ISLAMIC INFORMATION MANAGEMENT", "lecturer": "USTAZ DANIAL BIN ZUHAIR", "credit_hour": 2.0},
                {"code": "ELC231", "name": "INTEGRATED LANGUAGE SKILLS III", "lecturer": "SIR FARIS BIN FURQAN", "credit_hour": 3.0},
                {"code": "HBU131", "name": "NATIONAL KESATRIA III", "lecturer": "PUAN DHIA BINTI LOKMAN", "credit_hour": 1.0},
                {"code": "IML206", "name": "DIGITAL PRESERVATION IN LIBRARY ENVIRONMENT", "lecturer": "SIR HAFIZ BIN SHAMSUL", "credit_ho
                {"code": "IML207", "name": "INFORMATION SECURITY FOR LIBRARIES", "lecturer": "MISS FARHAH BINTI FARHAN", "credit_hour": 2
                {"code": "IML208", "name": "PROGRAMMING FOR LIBRARIES", "lecturer": "SIR NOAH BIN MUHAMMAD NAUFAL", "credit_hour": 4.0},
                {"code": "IML209", "name": "DESCRIPTIVE CATALOGING", "lecturer": "MADAM ZALIKHA BINTI SAAD", "credit_hour": 3.0},
            ],
```



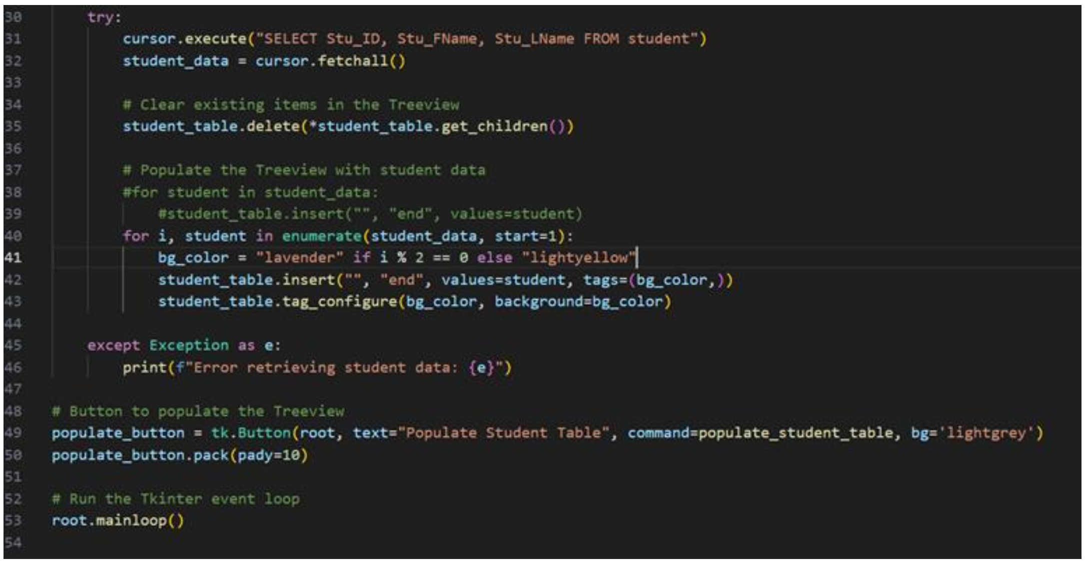
```
            "4": [
                {"code": "IMC258", "name": "METADATA DEVELOPMENT IN INFORMATION ENVIRONMENT", "lecturer": "DR. FAREHAH BINTI SULAIMAN", "
                {"code": "IML254", "name": "INTRODUCTION TO WEB CONTENT DEVELOPMENT", "lecturer": "PROF. DR. RIZMAN BIN SHAHRIL", "credit
                {"code": "IML255", "name": "SUBJECT CATALOGING AND CLASSIFICATION", "lecturer": "MADAM HUMAIRAH BINTI ZAHID", "credit_hou
                {"code": "IML256", "name": "MULTIMEDIA AND DIGITAL PUBLISHING IN LIBRARIES", "lecturer": "SIR HARRAZ BIN MALEEQ", "credit
                {"code": "IML257", "name": "LIBRARIES AND CUSTOMERS", "lecturer": "MADAM AMANI BINTI ROSLAN", "credit_hour": 3.0},
            ],
            "5": [
                {"code": "ENT300", "name": "FUNDAMENTALS OF ENTREPRENEURSHIP", "lecturer": "MADAM RAHIMAH BINTI YUSOF", "credit_hour": 3.
                {"code": "IML301", "name": "LIBRARY OUTREACH", "lecturer": "MISS SYAHIDA BINTI GHAZALI", "credit_hour": 4.0},
                {"code": "IML302", "name": "DIGITAL REFERENCE AND INFORMATION ANALYTICS", "lecturer": "DR. SYAFIQ BIN AHMAD", "credit_hou
                {"code": "IML303", "name": "INNOVATION IN LIBRARIES", "lecturer": "DR. HAKIM BIN MUHAMMAD SAMAD", "credit_hour": 3.0},
                {"code": "IML310", "name": "LIBRARY FIELDWORKS", "lecturer": "PROF. DR. ANNISA BINTI ABDUL MANAF", "credit_hour": 4.0},
            ],
        }

        return course_info.get(semester, [])

if __name__ == "__main__":
    root = tk.Tk()
    app = CourseGUI(root)
    root.mainloop()

```

3.3 SNAPSHOTS OF PYTHON CODE FOR TABLE GRADE_SEMESTER

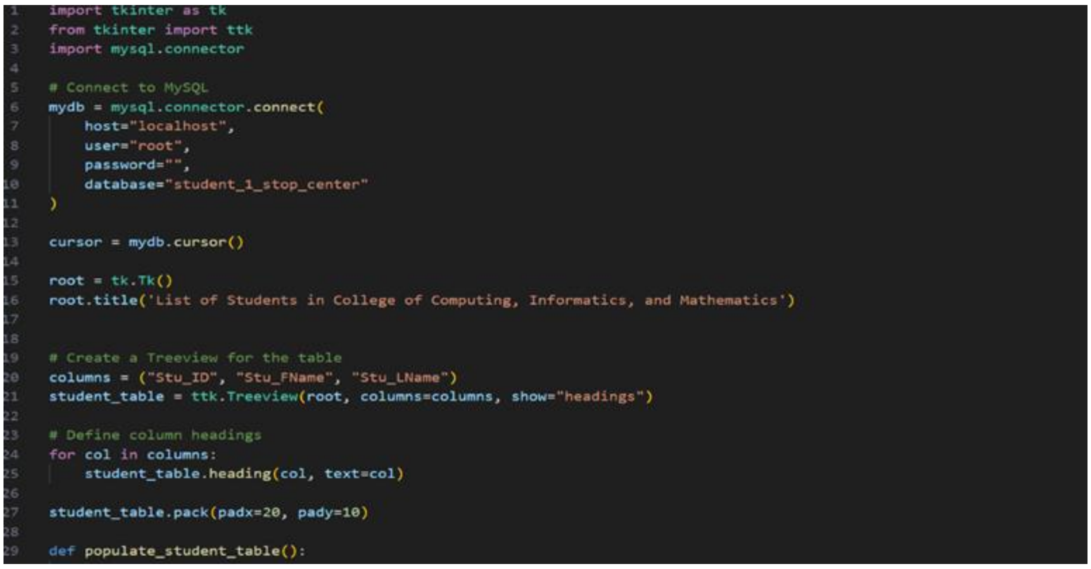


```
import tkinter as tk
from tkinter import messagebox
import mysql.connector

# Dictionary to map grades to grade points
grade_point_dict = {
    'A': 4.0,
    'A-': 3.7,
    'B+': 3.3,
    'B': 3.0,
    'B-': 2.7,
    'C+': 2.3,
    'C': 2.0,
    'C-': 1.7,
    'D+': 1.3,
    'D': 1.0,
    'F': 0.0
}

credit_hour_dict = {
    0.0: 0.0,
    1.0: 1.0,
    2.0: 2.0,
    3.0: 3.0,
    4.0: 4.0
```

3.4 SNAPSHOTS OF PYTHON CODE FOR STUDENT LIST

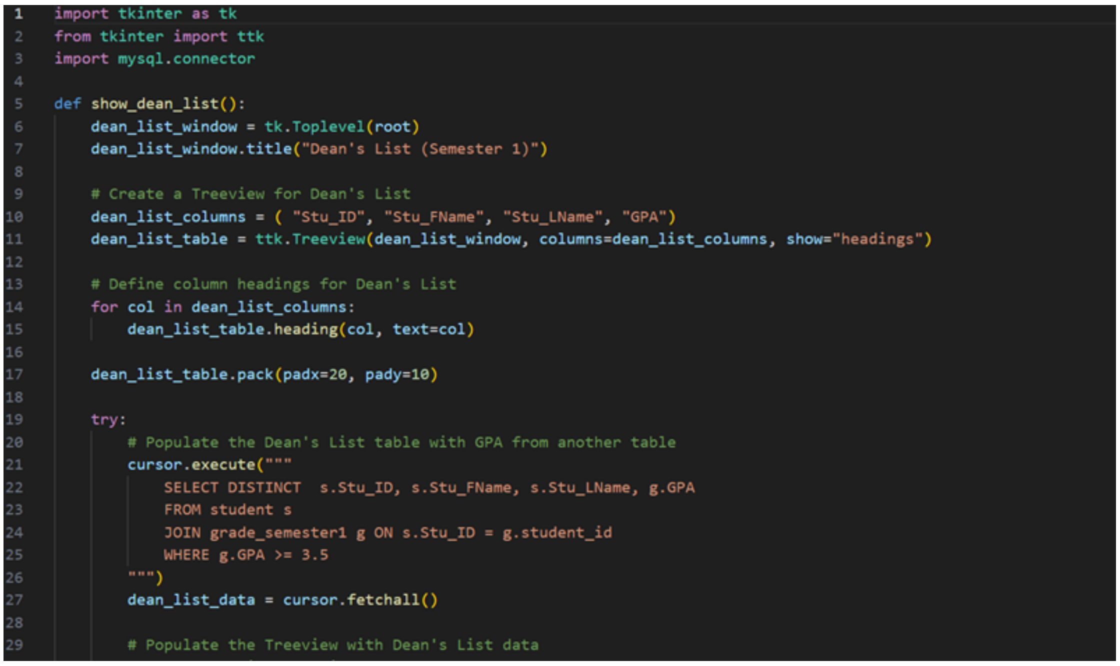


```python
    try:
        cursor.execute("SELECT Stu_ID, Stu_FName, Stu_LName FROM student")
        student_data = cursor.fetchall()

        # Clear existing items in the Treeview
        student_table.delete(*student_table.get_children())

        # Populate the Treeview with student data
        #for student in student_data:
            #student_table.insert("", "end", values=student)
        for i, student in enumerate(student_data, start=1):
            bg_color = "lavender" if i % 2 == 0 else "lightyellow"
            student_table.insert("", "end", values=student, tags=(bg_color,))
            student_table.tag_configure(bg_color, background=bg_color)

    except Exception as e:
        print(f"Error retrieving student data: {e}")

# Button to populate the Treeview
populate_button = tk.Button(root, text="Populate Student Table", command=populate_student_table, bg='lightgrey')
populate_button.pack(pady=10)

# Run the Tkinter event loop
root.mainloop()
```

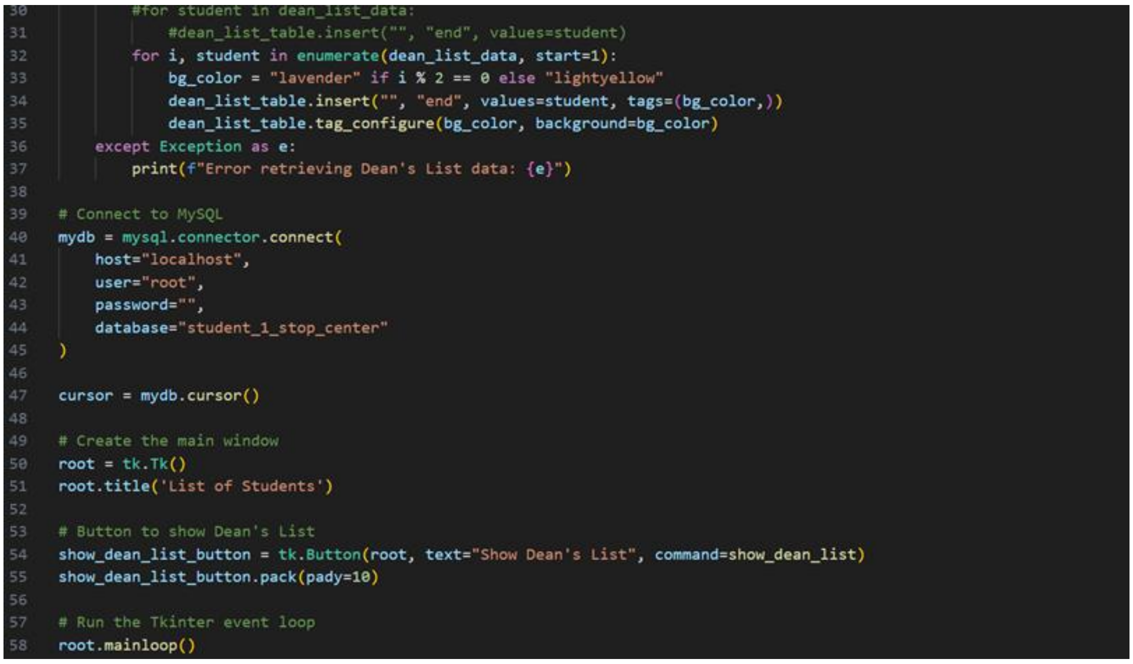


```python
import tkinter as tk
from tkinter import ttk
import mysql.connector

# Connect to MySQL
mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="student_1_stop_center"
)

cursor = mydb.cursor()

root = tk.Tk()
root.title('List of Students in College of Computing, Informatics, and Mathematics')


# Create a Treeview for the table
columns = ("Stu_ID", "Stu_FName", "Stu_LName")
student_table = ttk.Treeview(root, columns=columns, show="headings")

# Define column headings
for col in columns:
    student_table.heading(col, text=col)

student_table.pack(padx=20, pady=10)

def populate_student_table():
```

3.5 SNAPSHOTS OF PYTHON CODE FOR DEAN LIST

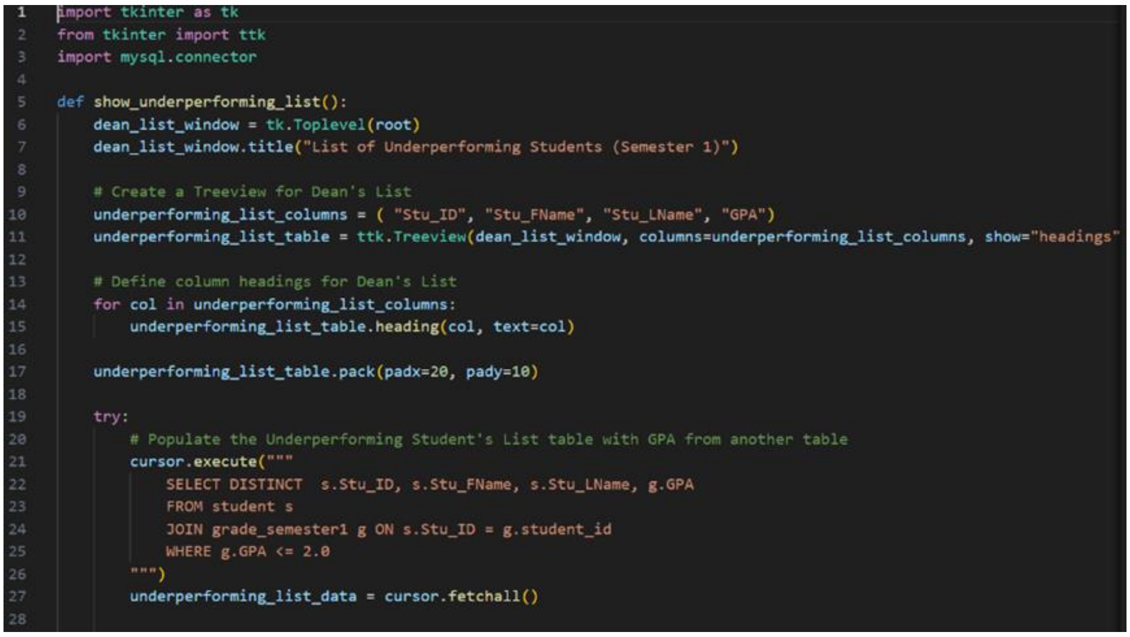


```python
import tkinter as tk
from tkinter import ttk
import mysql.connector

def show_dean_list():
    dean_list_window = tk.Toplevel(root)
    dean_list_window.title("Dean's List (Semester 1)")

    # Create a Treeview for Dean's List
    dean_list_columns = ( "Stu_ID", "Stu_FName", "Stu_LName", "GPA")
    dean_list_table = ttk.Treeview(dean_list_window, columns=dean_list_columns, show="headings")

    # Define column headings for Dean's List
    for col in dean_list_columns:
        dean_list_table.heading(col, text=col)

    dean_list_table.pack(padx=20, pady=10)

    try:
        # Populate the Dean's List table with GPA from another table
        cursor.execute("""
            SELECT DISTINCT  s.Stu_ID, s.Stu_FName, s.Stu_LName, g.GPA
            FROM student s
            JOIN grade_semester1 g ON s.Stu_ID = g.student_id
            WHERE g.GPA >= 3.5
        """)
        dean_list_data = cursor.fetchall()

        # Populate the Treeview with Dean's List data
```

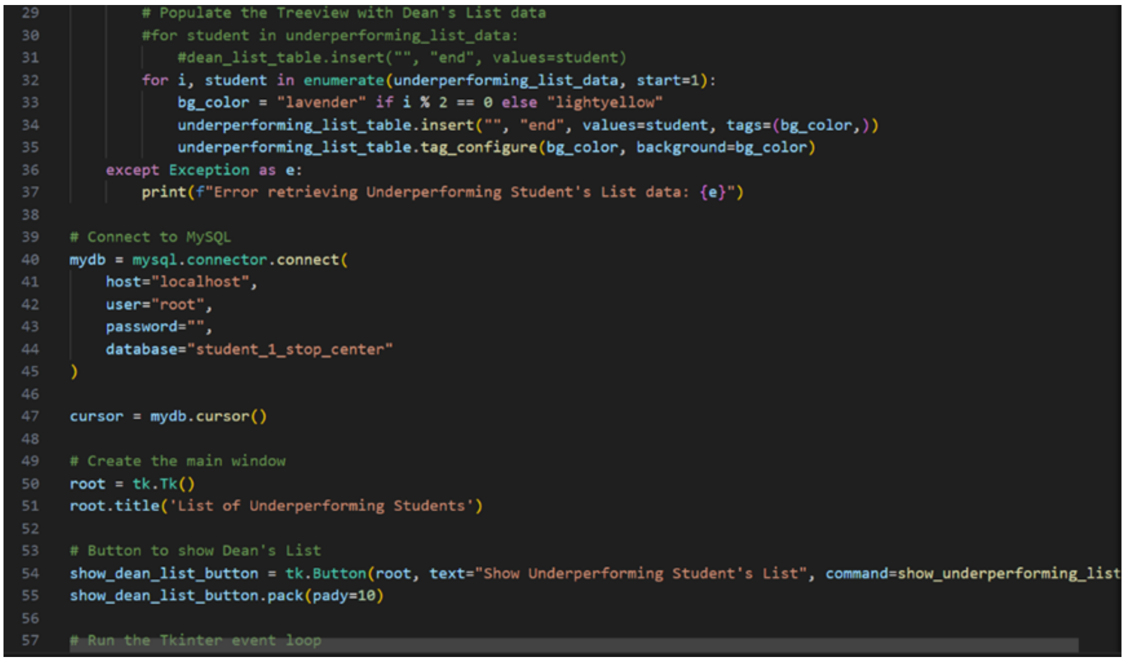


```python
        #for student in dean_list_data:
            #dean_list_table.insert("", "end", values=student)
        for i, student in enumerate(dean_list_data, start=1):
            bg_color = "lavender" if i % 2 == 0 else "lightyellow"
            dean_list_table.insert("", "end", values=student, tags=(bg_color,))
            dean_list_table.tag_configure(bg_color, background=bg_color)
    except Exception as e:
        print(f"Error retrieving Dean's List data: {e}")

# Connect to MySQL
mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="student_1_stop_center"
)

cursor = mydb.cursor()

# Create the main window
root = tk.Tk()
root.title('List of Students')

# Button to show Dean's List
show_dean_list_button = tk.Button(root, text="Show Dean's List", command=show_dean_list)
show_dean_list_button.pack(pady=10)

# Run the Tkinter event loop
root.mainloop()
```

3.6 SNAPSHOTS OF PYTHON CODE FOR UNDERPERFORMING STUDENTS LIST



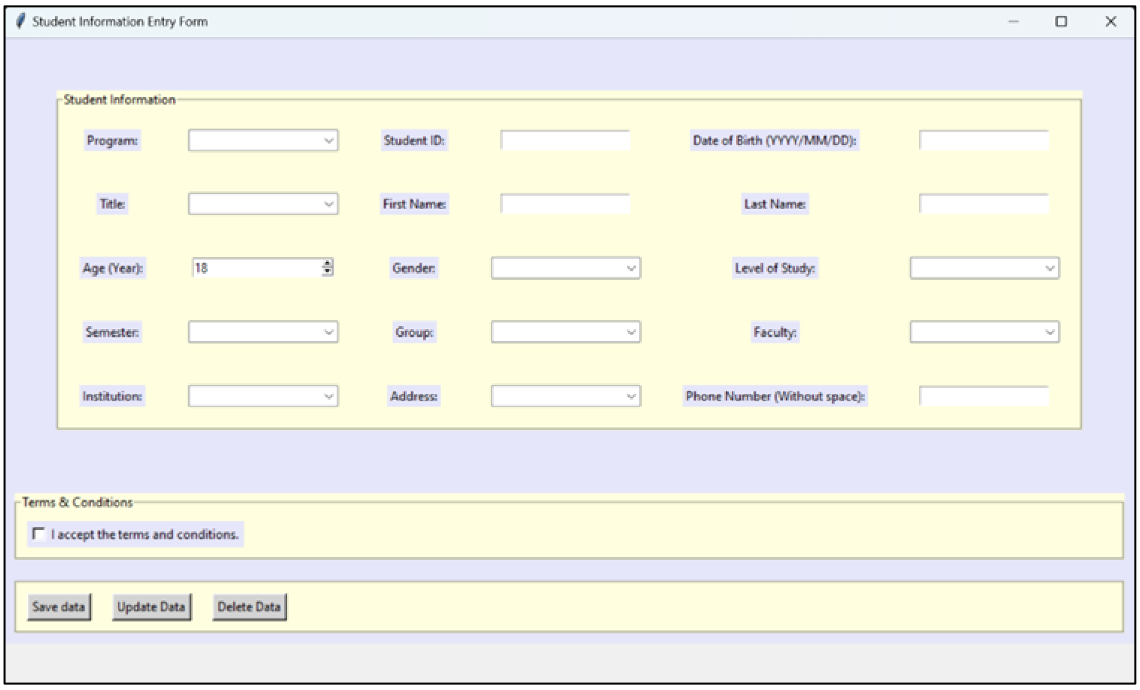
```python
import tkinter as tk
from tkinter import ttk
import mysql.connector

def show_underperforming_list():
    dean_list_window = tk.Toplevel(root)
    dean_list_window.title("List of Underperforming Students (Semester 1)")

    # Create a Treeview for Dean's List
    underperforming_list_columns = ( "Stu_ID", "Stu_FName", "Stu_LName", "GPA")
    underperforming_list_table = ttk.Treeview(dean_list_window, columns=underperforming_list_columns, show="headings"

    # Define column headings for Dean's List
    for col in underperforming_list_columns:
        underperforming_list_table.heading(col, text=col)

    underperforming_list_table.pack(padx=20, pady=10)

    try:
        # Populate the Underperforming Student's List table with GPA from another table
        cursor.execute("""
            SELECT DISTINCT  s.Stu_ID, s.Stu_FName, s.Stu_LName, g.GPA
            FROM student s
            JOIN grade_semester1 g ON s.Stu_ID = g.student_id
            WHERE g.GPA <= 2.0
        """)
        underperforming_list_data = cursor.fetchall()

```



```python
        # Populate the Treeview with Dean's List data
        #for student in underperforming_list_data:
            #dean_list_table.insert("", "end", values=student)
        for i, student in enumerate(underperforming_list_data, start=1):
            bg_color = "lavender" if i % 2 == 0 else "lightyellow"
            underperforming_list_table.insert("", "end", values=student, tags=(bg_color,))
            underperforming_list_table.tag_configure(bg_color, background=bg_color)
    except Exception as e:
        print(f"Error retrieving Underperforming Student's List data: {e}")

# Connect to MySQL
mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="student_1_stop_center"
)

cursor = mydb.cursor()

# Create the main window
root = tk.Tk()
root.title('List of Underperforming Students')

# Button to show Dean's List
show_dean_list_button = tk.Button(root, text="Show Underperforming Student's List", command=show_underperforming_list
show_dean_list_button.pack(pady=10)

# Run the Tkinter event loop
```

4.0 SNAPSHOTS OF GUI INTERFACE

4.1 SNAPSHOTS OF GUI INTERFACE FOR TABLE STUDENT


Student Information Entry Form

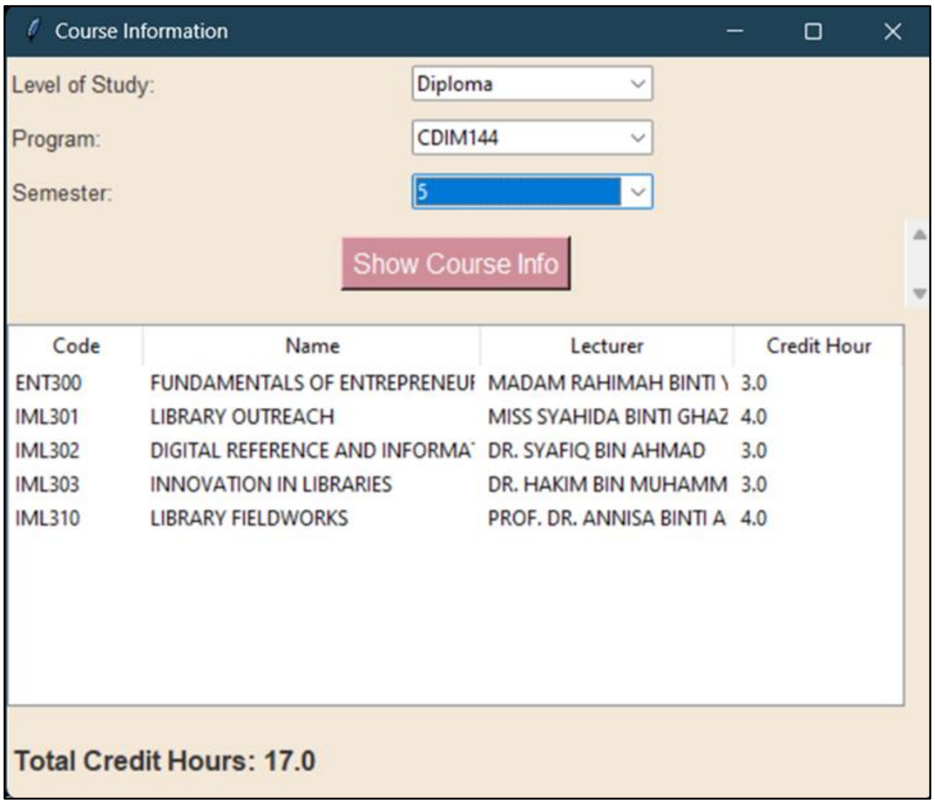
Student Information

Program:		Student ID:		Date of Birth (YYYY/MM/DD):	
Title:		First Name:		Last Name:	
Age (Year):	18	Gender:		Level of Study:	
Semester:		Group:		Faculty:	
Institution:		Address:		Phone Number (Without space):	

Terms & Conditions

☐ I accept the terms and conditions.

4.2 SNAPSHOTS OF GUI INTERFACE FOR TABLE COURSE


Course Information

Level of Study: Diploma

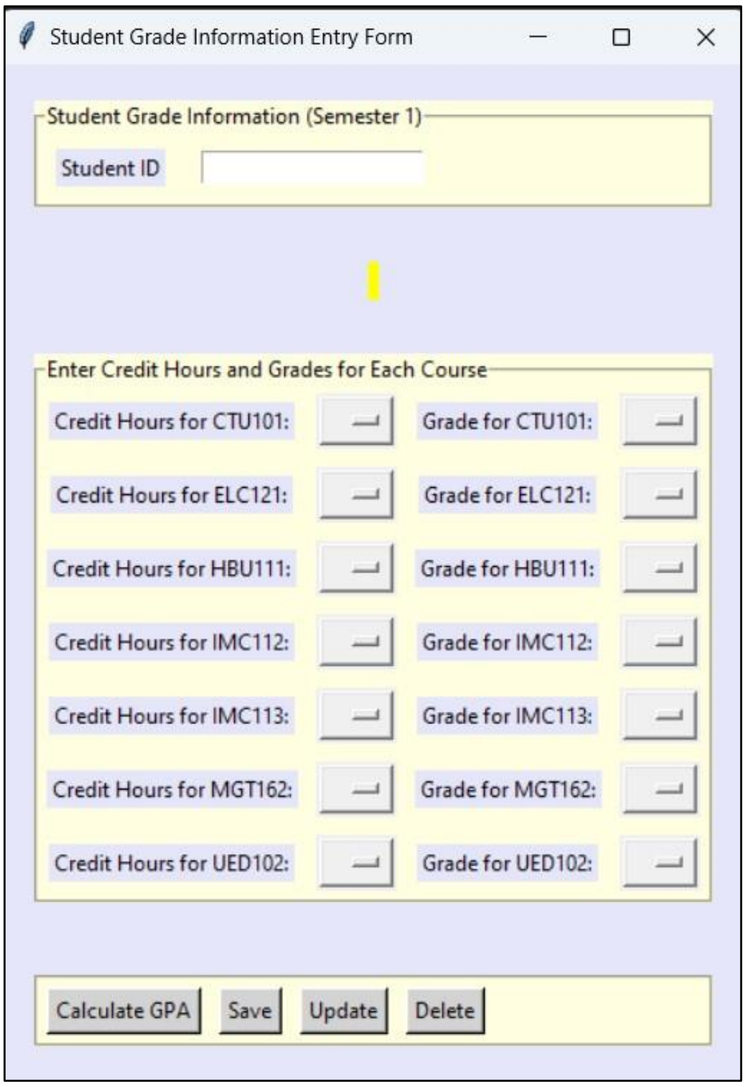
Program: CDIM144

Semester: 5

Code	Name	Lecturer	Credit Hour
ENT300	FUNDAMENTALS OF ENTREPRENEUI	MADAM RAHIMAH BINTI \	3.0
IML301	LIBRARY OUTREACH	MISS SYAHIDA BINTI GHAZ	4.0
IML302	DIGITAL REFERENCE AND INFORMA	DR. SYAFIQ BIN AHMAD	3.0
IML303	INNOVATION IN LIBRARIES	DR. HAKIM BIN MUHAMM	3.0
IML310	LIBRARY FIELDWORKS	PROF. DR. ANNISA BINTI A	4.0

Total Credit Hours: 17.0

4.3 SNAPSHOTS OF GUI INTERFACE FOR TABLE GRADE_SEMESTER


Student Grade Information Entry Form

Student Grade Information (Semester 1)

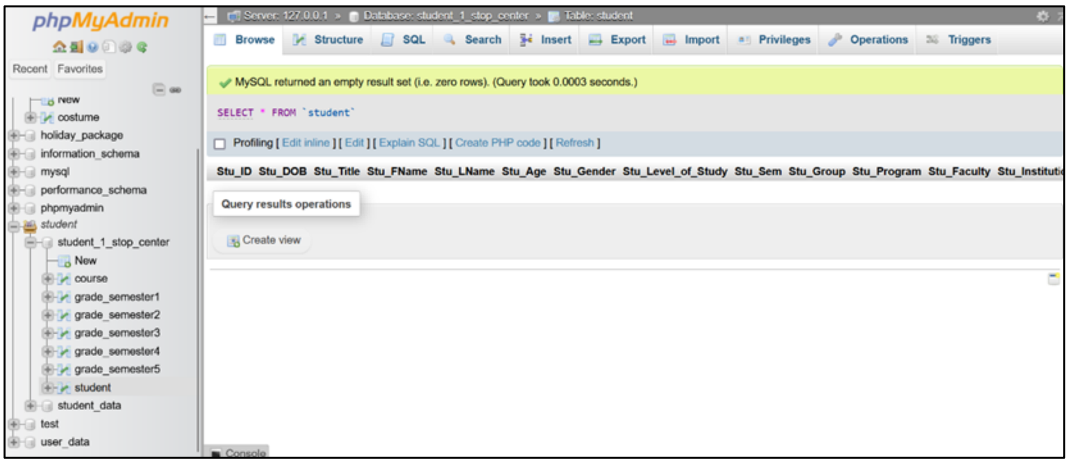
Student ID

Enter Credit Hours and Grades for Each Course

Credit Hours for CTU101:		Grade for CTU101:	
Credit Hours for ELC121:		Grade for ELC121:	
Credit Hours for HBU111:		Grade for HBU111:	
Credit Hours for IMC112:		Grade for IMC112:	
Credit Hours for IMC113:		Grade for IMC113:	
Credit Hours for MGT162:		Grade for MGT162:	
Credit Hours for UED102:		Grade for UED102:	

5.0 SNAPSHOTS OF DATABASE STUDENT 1 STOP CENTER

5.1 TABLE STUDENT



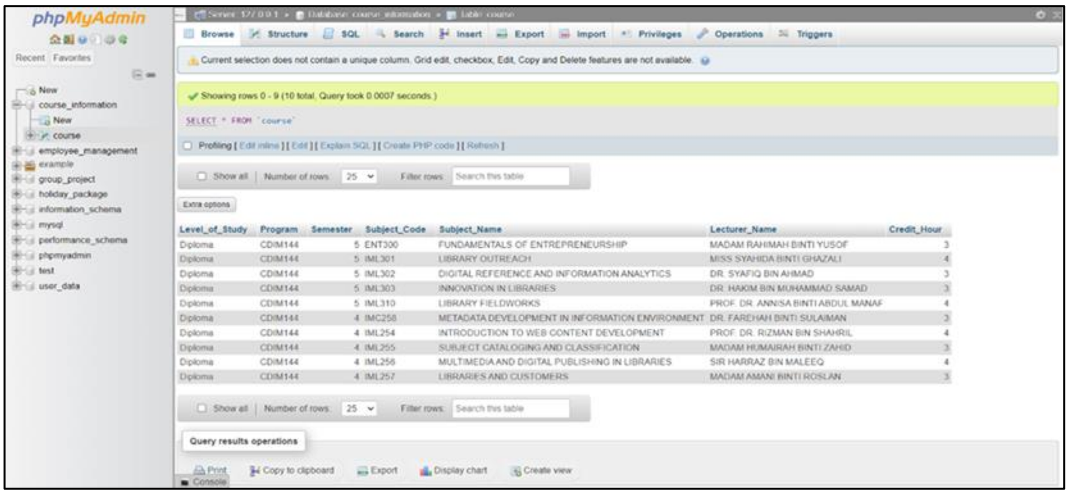
Server: 127.0.0.1 » Database: student_1_stop_center » Table: student

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0003 seconds.)

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

Stu_ID Stu_DOB Stu_Title Stu_FName Stu_LName Stu_Age Stu_Gender Stu_Level_of_Study Stu_Sem Stu_Group Stu_Program Stu_Faculty Stu_Instituti

5.2 TABLE COURSE



Server: 127.0.0.1 » Database: course_information » Table: course

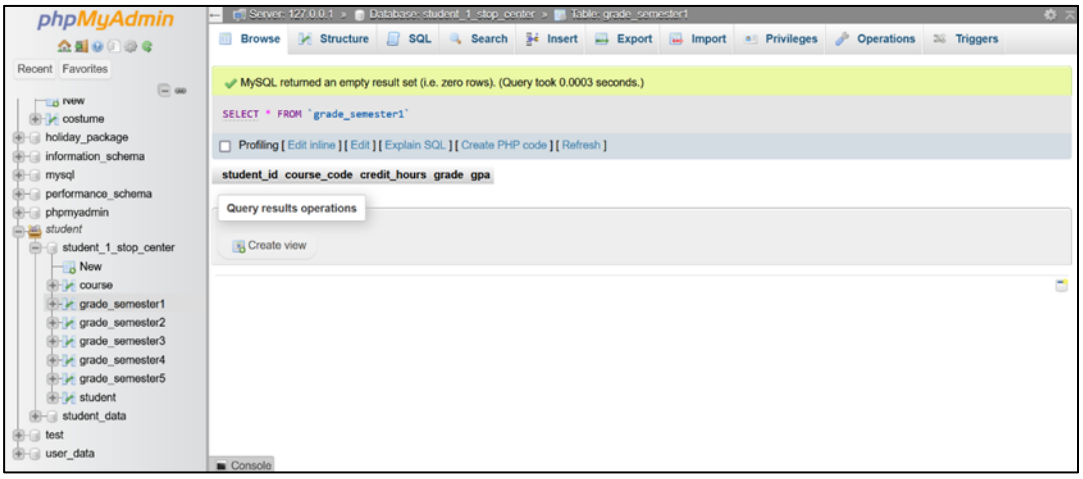
Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

Showing rows 0 - 9 (10 total, Query took 0.0007 seconds.)

```
SELECT * FROM `course`
```

Level_of_Study	Program	Semester	Subject_Code	Subject_Name	Lecturer_Name	Credit_Hour
Diploma	CDIM144	5	ENT300	FUNDAMENTALS OF ENTREPRENEURSHIP	MADAM RAHIMAH BINTI YUSOF	3
Diploma	CDIM144	5	IML301	LIBRARY OUTREACH	MISS SYAHIDA BINTI GHAZALI	4
Diploma	CDIM144	5	IML302	DIGITAL REFERENCE AND INFORMATION ANALYTICS	DR. SYAFIQ BIN AHMAD	3
Diploma	CDIM144	5	IML303	INNOVATION IN LIBRARIES	DR. HAKIM BIN MUHAMMAD SAMAD	3
Diploma	CDIM144	5	IML310	LIBRARY FIELDWORKS	PROF. DR. ANNISA BINTI ABDUL MANAF	4
Diploma	CDIM144	4	IMC258	METADATA DEVELOPMENT IN INFORMATION ENVIRONMENT	DR. FAREHAH BINTI SULAIMAN	3
Diploma	CDIM144	4	IML254	INTRODUCTION TO WEB CONTENT DEVELOPMENT	PROF. DR. RIZMAN BIN SHAHRIL	4
Diploma	CDIM144	4	IML255	SUBJECT CATALOGING AND CLASSIFICATION	MADAM HUMAIRAH BINTI ZAHID	3
Diploma	CDIM144	4	IML256	MULTIMEDIA AND DIGITAL PUBLISHING IN LIBRARIES	SIR HARRAZ BIN MALEEQ	4
Diploma	CDIM144	4	IML257	LIBRARIES AND CUSTOMERS	MADAM AMANI BINTI ROSLAN	3

5.3 TABLE GRADE_SEMESTER



Server: 127.0.0.1 » Database: student_1_stop_center » Table: grade_semester1

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0003 seconds.)

```
SELECT * FROM `grade_semester1`
```

student_id course_code credit_hours grade gpa

6.0 CONCLUSION

The group project provided us with the opportunity to develop a system for calculating the grade point average (GPA) based on credit hours and courses. Despite facing challenges, the project was a valuable experience that allowed us to work together as a team. Our system ensures the accuracy and reliability of GPA calculations, as well as the validation of credit hours. Additionally, it features a user-friendly interface with clear instructions for accessibility.

To calculate the GPA, the basic formula is to divide the total points earned in a program by the total number of credits attempted. The resulting figure is the GPA for that program. The point values for letter grades are typically used to calculate the GPA, with each grade assigned a specific point value,

The GPA calculation is an important aspect of academic achievement, and our system aims to provide an accurate and reliable method for students to determine their GPA.

STUDENT PLEDGE OF ACADEMIC INTEGRITY

As a student of Universiti Teknologi MARA (UiTM), it is my responsibility to act in accordance with UiTM's academic assessment and evaluation policy. I hereby pledge to act and uphold academic integrity and pursue scholarly activities in UiTM with honesty and responsible manner. I will not engage or tolerate acts of academic dishonesty, academic misconduct, or academic fraud including but not limited to:

a. **Cheating:** Using or attempt to use any unauthorized device, assistance, sources, practice or materials while completing academic assessments. This include but not limited to copying from another, allowing others to copy, unauthorized collaboration on an assignment or open book tests, or engaging in any act or conduct that can be construed as cheating.
b. **Plagiarism:** Using or attempts to use the work of others (ideas, design, words, art, music, etc.) without acknowledging the source; using or purchasing materials prepared by another person or agency or engaging in other behavior that a reasonable person would consider as plagiarism.
c. **Fabrication:** Falsifying data, information, or citations in any academic assessment and evaluation.
d. **Deception:** Providing false information with intend to deceive an instructor concerning any academic assessment and evaluation.
e. **Furnishing false information:** Providing false information or false representation to any UiTM official, instructor, or office.

With this pledge, I am fully aware that I am obliged to conduct myself with utmost honesty and integrity. I fully understand that a disciplinary action can be taken against me if I, in any manner, violate this pledge.

Name : NUR MAISARAH BINTI ABDUL MANAH
Matric Number : 2022892948
Course Code : IML209
Programme Code :-
Faculty / Campus : UiTM Kampus Sungai Petani

UNIVERSITI TEKNOLOGI MARA

STUDENT PLEDGE OF ACADEMIC INTEGRITY

As a student of Universiti Teknologi MARA (UiTM), it is my responsibility to act in accordance with UiTM's academic assessment and evaluation policy. I hereby pledge to act and uphold academic integrity and pursue scholarly activities in UiTM with honesty and responsible manner. I will not engage or tolerate acts of academic dishonesty, academic misconduct, or academic fraud including but not limited to:

a. **Cheating:** Using or attempt to use any unauthorized device, assistance, sources, practice or materials while completing academic assessments. This include but not limited to copying from another, allowing others to copy, unauthorized collaboration on an assignment or open book tests, or engaging in any act or conduct that can be construed as cheating.
b. **Plagiarism:** Using or attempts to use the work of others (ideas, design, words, art, music, etc.) without acknowledging the source; using or purchasing materials prepared by another person or agency or engaging in other behavior that a reasonable person would consider as plagiarism.
c. **Fabrication:** Falsifying data, information, or citations in any academic assessment and evaluation.
d. **Deception:** Providing false information with intend to deceive an instructor concerning any academic assessment and evaluation.
e. **Furnishing false information:** Providing false information or false representation to any UiTM official, instructor, or office.

With this pledge, I am fully aware that I am obliged to conduct myself with utmost honesty and integrity. I fully understand that a disciplinary action can be taken against me if I, in any manner, violate this pledge.

Name : PRISCILLA ANN VINCENT
Matric Number : 2022679716
Course Code : IML208
Programme Code :-
Faculty / Campus : UiTM Kampus Sungai Petani

STUDENT PLEDGE OF ACADEMIC INTEGRITY

As a student of Universiti Teknologi MARA (UiTM), it is my responsibility to act in accordance with UiTM's academic assessment and evaluation policy. I hereby pledge to act and uphold academic integrity and pursue scholarly activities in UiTM with honesty and responsible manner. I will not engage or tolerate acts of academic dishonesty, academic misconduct, or academic fraud including but not limited to:

a. **Cheating:** Using or attempt to use any unauthorized device, assistance, sources, practice or materials while completing academic assessments. This include but not limited to copying from another, allowing others to copy, unauthorized collaboration on an assignment or open book tests, or engaging in any act or conduct that can be construed as cheating.
b. **Plagiarism:** Using or attempts to use the work of others (ideas, design, words, art, music, etc.) without acknowledging the source; using or purchasing materials prepared by another person or agency or engaging in other behavior that a reasonable person would consider as plagiarism.
c. **Fabrication:** Falsifying data, information, or citations in any academic assessment and evaluation.
d. **Deception:** Providing false information with intend to deceive an instructor concerning any academic assessment and evaluation.
e. **Furnishing false information:** Providing false information or false representation to any UiTM official, instructor, or office.

With this pledge, I am fully aware that I am obliged to conduct myself with utmost honesty and integrity. I fully understand that a disciplinary action can be taken against me if I, in any manner, violate this pledge.

Name : NOOR HASMURNI BINTI SHAKRI
Matric Number : 2022449098
Course Code : IML208
Programme Code :-
Faculty / Campus : UiTM Kampus Sungai Petani

STUDENT PLEDGE OF ACADEMIC INTEGRITY

As a student of Universiti Teknologi MARA (UiTM), it is my responsibility to act in accordance with UiTM's academic assessment and evaluation policy. I hereby pledge to act and uphold academic integrity and pursue scholarly activities in UiTM with honesty and responsible manner. I will not engage or tolerate acts of academic dishonesty, academic misconduct, or academic fraud including but not limited to:

a. **Cheating:** Using or attempt to use any unauthorized device, assistance, sources, practice or materials while completing academic assessments. This include but not limited to copying from another, allowing others to copy, unauthorized collaboration on an assignment or open book tests, or engaging in any act or conduct that can be construed as cheating.
b. **Plagiarism:** Using or attempts to use the work of others (ideas, design, words, art, music, etc.) without acknowledging the source; using or purchasing materials prepared by another person or agency or engaging in other behavior that a reasonable person would consider as plagiarism.
c. **Fabrication:** Falsifying data, information, or citations in any academic assessment and evaluation.
d. **Deception:** Providing false information with intend to deceive an instructor concerning any academic assessment and evaluation.
e. **Furnishing false information:** Providing false information or false representation to any UiTM official, instructor, or office.

With this pledge, I am fully aware that I am obliged to conduct myself with utmost honesty and integrity. I fully understand that a disciplinary action can be taken against me if I, in any manner, violate this pledge.

Name : NURUL ANIS ADEELA BINTI ROSELE
Matric Number : 2022606634
Course Code : IML208
Programme Code :-
Faculty / Campus : UiTM Kampus Sungai Petani

FAKULTI PENGURUSAN MAKLUMAT
UNIVERSITI TEKNOLOGI MARA

<u>Project Evaluation Scheme</u>

<u>Group members:</u>

1. ……………………………………………………………………………………………
2. ……………………………………………………………………………………………
3. ……………………………………………………………………………………………
4. ……………………………………………………………………………………………
5. ……………………………………………………………………………………………

Program	IM110	**Course**	IMD238

	Descriptions	**Marks**
1.	**Correct program result:** **8-10: Excellence | 5-7: Good: 2-4: | Poor: | 1: Fatal**	**/40**
	• Program runs without error (10) • Handles incorrect input data (10) • Display appropriate error messages (10) • Calculate correct results (10)	
2.	**Programming style:** **4 - 5: Excellence | 2-3: Good: | 1: Poor**	**/10**
	• Sufficient comments to allows the program to be easily understood (5) • Easy to read code & Indentation (5)	
3.	**Design:** **8-10: Excellence | 5-7: Good: 2-4: | Poor: | 1: Very Poor**	**/20**
	• <u>Interface 10):</u> Adequate user prompting/user friendly (easy to navigate) Use of appropriate color and graphics • <u>Coding / solution (10):</u> Understandable variable names. (not just x,y,etc.) Use of functions to enhance readability Use array, decision and repetition structures where necessary	
4.	**Report:** **4 - 5: Excellence | 2-3: Good: | 1: Poor | 0: None**	**/15**
	• Introduction, objective and problem statement stated clearly (5) • Flowchart/pseudocode - using correct symbols/statements (5) • Printout of source code and form design (5)	
5.	**Bonus:** **4 - 5: Excellence | 2-3: Good: | 1: Poor**	**/5**
	Extra effort by students to make project interesting and attractive (5)	
6.	**Presentation:** **8-10: Excellence | 5-7: Good: 2-4: | Poor: | 1: Very Poor**	**/10**
	• Group participation in presentation • Clear explanation • Answer questions confidently	
	Total	**100%**