

FORMAT AND GUIDELINES TO PREPARE PROJECT REPORT (FOR BOTH UG & PG)

The following sequence of topics to be maintained in the Project Report:

Title Page

Bonafide Certificate

Bonafide from the place where the work is carried out, on their letter head

Declaration

Acknowledgement – Order of names to be acknowledged (refer page no. 4 for hierarchy)

Abstract of the work

Contents with page numbers

List of Abbreviations, if applicable

List of figures, list of tables, if applicable

Chapter – 1, Introduction

Chapter – 2, Aim and Scope of the present investigation

Chapter – 3, Experimental or materials and methods; algorithms

used **Chapter – 4**, Results and Discussion, Performance Analysis

Chapter – 5, Summary and Conclusions

References (in the following order only)

- for Journals – name of authors, journal name, volume number, year of publication, page numbers - starting and ending.
- For Books – name of author, title, name of editor, publisher, year, page numbers.
- Valid internet articles.

Sub chapters should be numbered as follows:

Eg. Sub Chapter of chapter 1 should be numbered as Chapter 1.1

Further sub chapters should be numbered as 1.1.1, 1.1.2 ,....

Letter Font and Line spacings:

To be followed throughout the project work

Line space – 1.5

Font Size - 12

Font Type - Arial

Margin space - Left 3cm; Right – 2.5cm; Top- 2.5cm; and Bottom – 2.5 cm

Major headings – all caps – 14 font size

Sub headings – all caps – 12 font size

Further sub headings – 12 font size, bold, italics

Still further sub headings – 12 font size, italics

Legend for Figures & tables: (Numbering should start with each chapter Number)

Figure – Fig:1.1 : Self descriptive title, 12 font size, italics, bold (should be below the figure)

Table – Table 1.1 : self descriptive title, 12 font size, bold (should be above the table)

Note: Fig 1.1 refers to Figure 1 of chapter 1, Fig 2.2 refers to fig 2 of Chapter 2

Note : Table 1.1 refers to Table 1 of Chapter 1, Table 3.2 refers to Table 2 of Chapter 3.

Page numbers should be centered in the number format 1,2,3..... at the bottom of each page, starting from Introduction chapter onwards.

For Abstract, Contents, List of Abbreviations, List of figures, list of tables – the page numbers should be in the Roman Number format (i), (ii), (iii)..... and centered at the bottom of the page.

PROFESSIONAL TRAINING REPORT
at
Sathyabama Institute of Science and Technology
(Deemed to be University)

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Technology Degree in

Information Technology

By

MOHAMMED SALMAN (37120047)



DEPARTMENT OF INFORMATION TECHNOLOGY
SCHOOL OF COMPUTING
SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119. TAMIL NADU

AUGUST 2020



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with "A" Grade by NAAC
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai - 600 119.

Phone: 044 - 2450 3150 / 51 / 52 / 54 / 55 Fax: 044 - 2450 2344
www.sathyabama.ac.in



SCHOOL OF COMPUTING

BONAFIDE CERTIFICATE

This is to certify that this Professional Training Report is the bonafide work of **MOHAMMED SALMAN (37120047)** who underwent the professional training in "**COLLEGE MANAGEMENT SYSTEM USING PYTHON**" under our supervision from July 2020 to September 2020.

Internal Guide

Dr.R.SUBHASHINI, M.E.,Ph.D.,

Head of the Department

Dr. R.SUBHASHINI, M.E.,Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, MOHAMMED SALMAN (37120047) hereby declare that the Professional Training Report on “**COLLEGE MANAGEMENT SYSTEM USING PYTHON**” done by me under the guidance of **Dr. R.SUBHASHINI, M.E.,Ph.D.**, at Sathyabama Institute Of Science And Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in Information Technology.

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind Encouragement in doing this project and for completing it successfully. I am grateful to them

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D, Dean**, School of Computing , **Dr.SUBHASHINI M.E., Ph.D. ,** Head of the Department of Information and Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.SUBHASHINI M.E.,Ph.D.,**for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

ABSTRACT

Main aim in developing College Management System is to provide an easy way not only to automate all functionalities of a college, but also to provide full functional reports to top management of college with the finest of details about any aspect of college.

College Management System is software has the perspective of attaining attraction of those colleges which don't have one good performing software for keeping their information secure and make their management easier. College Management System provides one attractive environment where you can manipulate data and information about students and staff easily. So we can say the Core purpose of designing "College Management System" is to manage the task related to the college students/employees and to reduce time to searching of appropriate candidates in college view.

This system provides the detail structure of the college campus and its departments. College Management System synchronizes the working of all the departments. It looks on all aspects of a college, its students, faculties, Departments, marks and other co – curricular activities.

TRAINING CERTIFICATE



To Whomsoever It May Concern,

This is to certify that **MOHAMMED SALMAN** has worked at our organization Haso Technology from **03rd February to 31st July, 2020** as Python Programmer Intern.

He has worked on Python Programmer and is capable of designing and promotion to cater to diverse needs. His work reflects his thoroughness in understanding and implementing his knowledge on python programmer platforms. His communication skill and leadership qualities and his thorough knowledge on python programmer are his most valuable assets .

We wish his good luck and success in his future endeavors.

Sincerely,

Dilip Kumar K K
HasoTechnology

ONLINE CERTIFICATE



LIST OF ABBREVATIONS

CHAPTER NO	TITLE	PAGE NO
1	INTRODUCTION	1
	1.1 OUTLINE OF THE PROJECT	1
	1.2 LITERATURE VIEW	1
	1.3 SYSTEM IMPLEMENTATION	2
	1.4 OBJECTIVES	3
2	AIM AND SCOPE OF THE PRESENT INVESTIGATION	4
	2.1 COLLEGE MANAGEMENT	4
	2.2 SERVER PORTAL	4
3	EXPERIMENTAL OR MATERIALS AND METHODS ALGORITHM USED	5
	3.1 REQUIREMENTS SPECIFICATION	5
	3.1.1 HARDWARE REQUIREMENTS	5
	3.1.2 SOFTWARE REQUIREMENTS	5
	3.2 ABOUT THE SOFTWARE	
	3.2.1 PYCHARM	
	3.2.2 MYSQL	
4.	RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS	6
	4.1 OVER VIEW OF THE PLATFORM	6
	4.2 TESTING	6
5	SUMMARY AND CONCLUSIONS	
	5.1 CONCLUSION	8
	5.2 FURTHER ENHANCEMENT	9

6	REFERENCES	10
7	APPENDIX	11
	a. SAMPLE CODE	
	b. SCREENSHOTS	

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO
1	ER DIAGRAM	3
2	BLOCK DIAGRAM	4
3	ADMIN BLOCK	7

1.INTRODUCTION

1.1 OUTLINE OF THE PROJECT

College Management System for college students have been developed which aims to rate and analyse the college faculty's performance. This type of College Management System reduces , the strenuous work of physically examining the all pages of each and every student. The system also reduces the burden of efforts and burden of keeping and maintaining the records on a manual base, it requires a lot of space and safety to keep up such records. Also the College Management can be tempered for wrong reasons in case of paper based data wherein the SFS will always ensure safety of data privacy. Another important features of the SFS is that physical presence of neither the admin nor the student is required for the either giving the data nor for assessing the data . Also further enhancement can be done and more features can be added for better retrieval of the data details.

1.2 LITERATURE VIEW

The College Management System information system for education establishments to manage student data. College Management System provide capabilities for selecting particular subject of data and generate the report automatically, build student details, student-related data needs in a college. Using the latest Microsoft technology, the .Net framework and ASP.Net, our system is a complete College Management System and staff interaction system with built-in features proofing that easily supports Integration with the latest student staff feedback system providers. A College Management System is an automatic data generation system that provides the proper data to the teachers as per the categories like always, poor, usually, very often, sometimes. Download Student data details Report .The provision may as well have a database of survey which can be given to the understudies following a secured. By this process student can see their data in online system whenever they need. This system is designed to be simple to use, simple to understand and easy to implement and configure to fit. After giving feedback by every student papers are collected by the faculty and calculated the overall grade for each subject and each lecturer. After that those all grade report is viewed by the HOD which is given by the faculty. It is provided as an Application Service Provision thus offering low setup

costs. College Management System.doc (Size: 2.3 MB / Downloads: 858). Hence estimating the performance of lecturers and giving feedback to college staff. So, the existing system carries more time to do a piece of work for this reason the College Management System is implemented.

1.3 SYSTEM IMPLEMENTATION

In this project we implemented the system which works on the concept of College Management System from students which is the replacement of existing paper data system which is so complicated method for finding an average for particular subject data which is submitted by faculty. This project has two kinds of users Student, Staff. In this system we have design main three forms student login form, admin login form, faculty login form. Student will login in their login form with proper id and password which is provided by admin at the time of registration. The student can see marks in online system provided by college staff. First of staff can prepare questions and add, update these questions to the online system. After that if student want any changes in his profile then there is an option for edit such particular information of his own. After that the subject contains page is displayed, where the subject is located in it and as per the selection from student students continued to their data after this page the mark page is displayed. This system can be developed using After that it was viewed by the students and can see their data about the lecturers. This page contains a form which includes such questions about the teachers where it contains some simple question regarding particular subject faculty. After seeing the data the page appeared which contains a message like you are data is seen. Then after data is seeing there is a log out button after click on it student will successfully log out from their profile. In main page it contains some hyperlinks like contact us, frequently ask questions, and submit the queries. And contains three particular modules like student login, admin login, staff login etc. Admin will fill the information of students as well as view their feedback, add and delete student and generate the average report of that feedback. After that log out button occurs and admin simply logout from the system. Student will login by using user id and password and after the login page will be open contains subjects, suggestion box, submit suggestions, submit feedback and after their occurs a log out button for logout successfully from the system.

1.4 OBJECTIVES

- 1) Decision making power is provided by this system.
- 2) Accurate result can be obtained.
- 3) This system makes Selection process more effective
- 4) To increase efficiency proposed system is depend on classification method.
- 5) Proposed system is used to reduce confusion at the time of processing data average

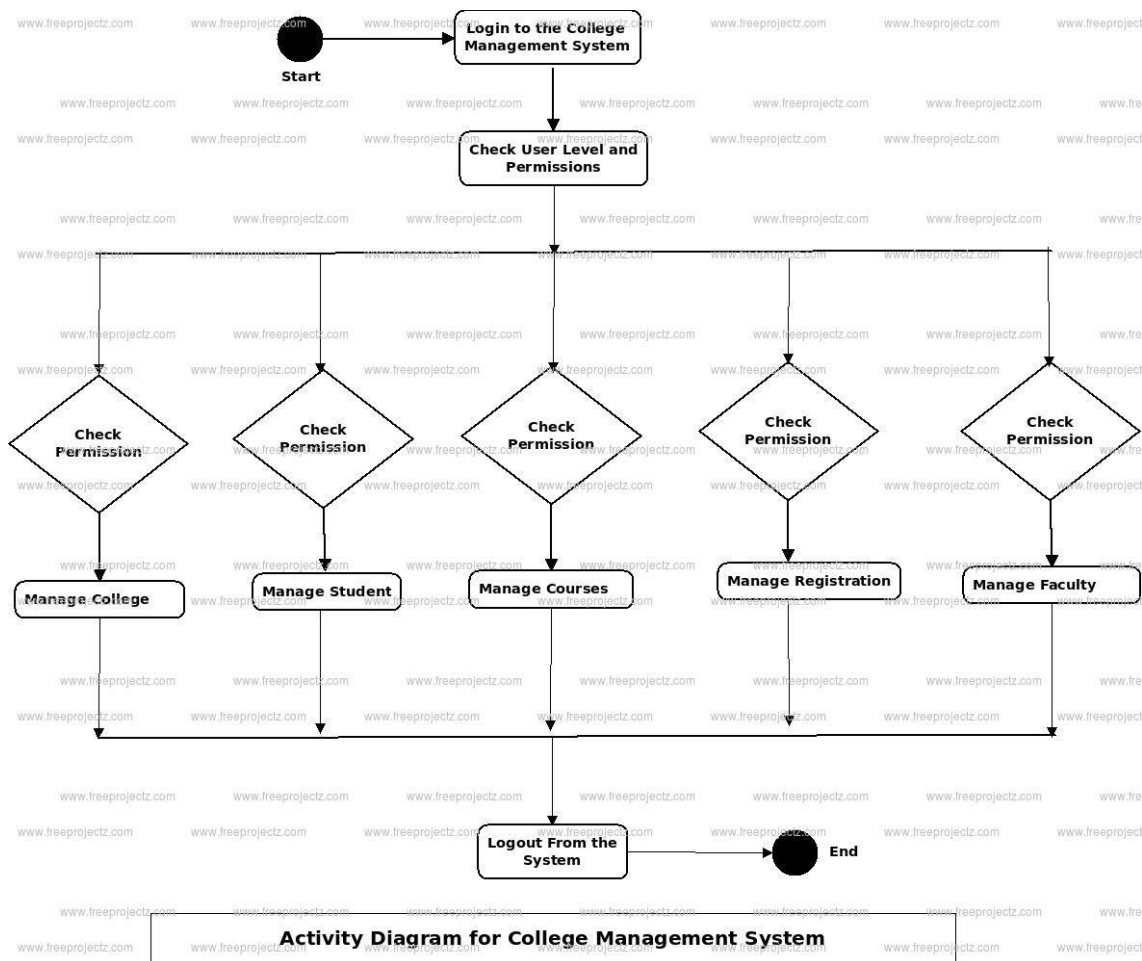


Figure: 1

2. AIM AND SCOPE OF THE PRESENT INVESTIGATION

2.1 COLLEGE MANAGEMENT SYSTEM

Finally, report was referred by the faculty and give suggestions to students to improve their learning skills. These are the four modules present in the project so we have developed these things in very easy and clearly understandable way. So, by seen above description the proposed system having many advantages as compared to the existing system.

2.2 SERVER PORTAL

A server portal is a commonly used phrase to describe the page where admin can have a user name and password to gain access to an education organization's programs and other learning related materials.

For example, a learner who has enrolled in an online certification program may use a student portal to access online course materials, such as articles, lectures and videos, hosted on the school's servers. Student portals may be used to provide information about the school, special events, course details, calendars, academic resources and contact information.

Some student portals may integrate with other systems to provide students with easy access to personal information, such as as tracking bursary awards and campus medical insurance

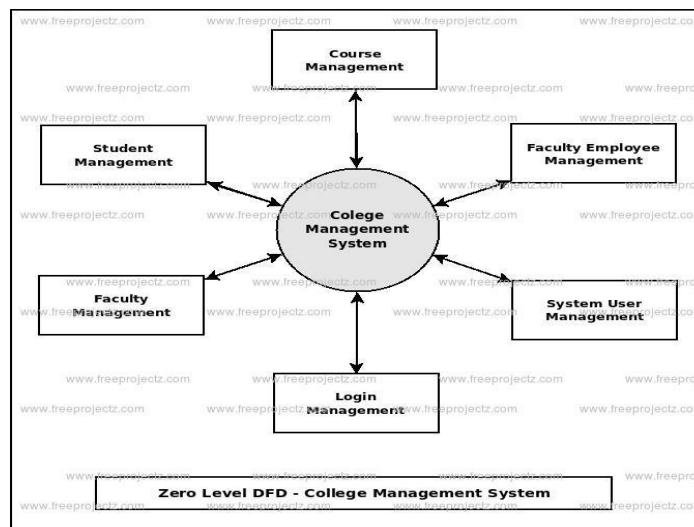


Figure:2

3. EXPERIMENTAL OR MATERIALS AND METHODS

3.1 REQUIREMENTS SPECIFICATION

3.1.1 HARDWARE REQUIREMENTS

Processor	- Intel CORE i5
Ram	- 4GB Ram
Hard disk drive	- 40GB
Monitor	-LCD

3.1.2 SOFTWARE REQUIREMENTS

Operating System	- WINDOWS XP
Software	- PYCHARM, XWAMP, WAMP

3.2 ABOUT THE SOFTWARE

3.2.1 PYCHARM

PyCharm is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains.^[6] It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django as well as data science with Anaconda.

WHY PYCHARM ?

1. PyCharm provides an API so that developers can write their own plugins to extend PyCharm features.
2. Several plugins from other JetBrains IDE also work with PyCharm.
3. There are more than 1000 plugins which are compatible PyCharm

3.2.2 MYSQL

Mysql is an Oracle-backed open source relational database management system RDBMS based on Structured Query Language SQL. Mysql runs on virtually all platforms, including Linux, Unix and Windows. Although it can be used in a wide range of application.

WHY MYSQL ?

1. Data Security
2. On-Demand Scalability
3. High Performance
4. The Flexibility of Open Source

4. RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS

4.1 OVER VIEW OF THE PLATFORM

The user can log into the system with a valid user ID and password; fill in an college management form and see the data to the system. The administrator can analyze the data. The college management system is platform independent. So there is no need for installing any additional software on the client systems. The new system issues the users, asset of questionnaires. The users answer the question and see the data. This is very effective, fast and cost effective method for see the data. The users who see the data are authenticated with a login ID and password

4.2 TESTING

Software testing is a critical element of software quality assurance and represents the ultimate reuse of specification. Design and code testing represents interesting anomaly for the software during earlier definition and development phase, it was attempted to build software from an abstract concept to tangible implementation. The testing phase involves, testing of the development of the system using various techniques such as White Box Testing Control Structure Testing.

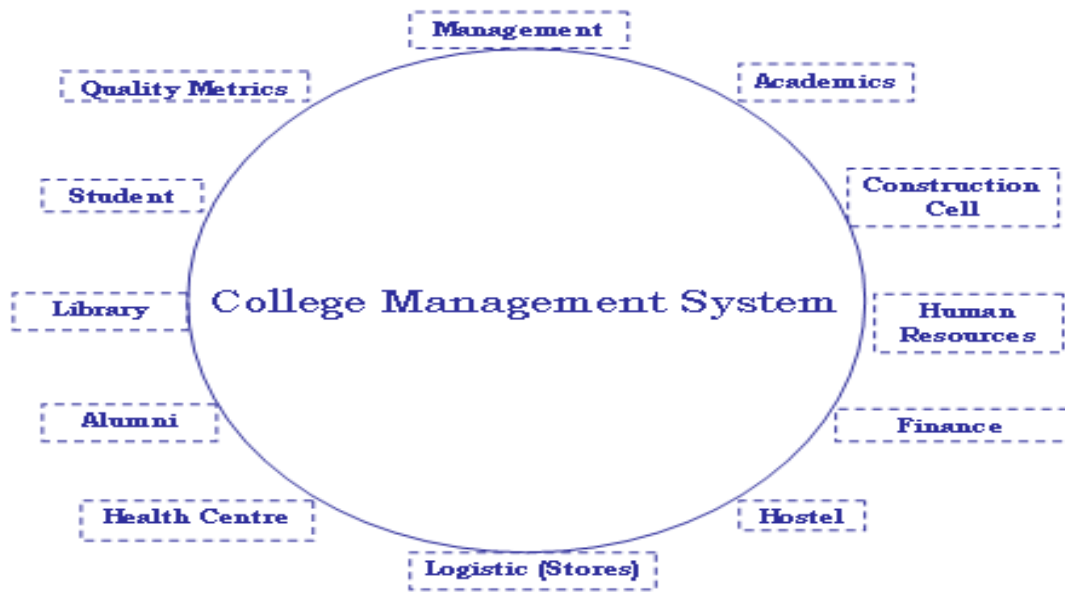


Figure 3

5. SUMMARY AND CONCLUSIONS

5.1 CONCLUSION

This project “college management system using python” we proposed a uniform identification solution. We also developed server oriented project can be easily applied to any network our methods is scalable and can be easily applied on online application.

5.2 FURTHER ENHANCEMENT

In this project, we have added all the future enhancement for designing and development the page all the users cant access the service at any time if the page will come the will be act as a friendly for the users.

5.3 REFERENCES

1. <https://forums.mysql.com>
2. <https://www.quora.com>
3. <https://tutorialpoint.com>
4. <https://www.w3school.net>
5. <https://www.jetbrains.com/pycharm/>

APPENDIX

a. Sample Code

1.menu

```
from tkinter import*
import random
import os

def __marksheet__():
    filename = 'Search_Page.py'
    os.system(filename)
    os.system('notepad'+filename)

def __Library__():
    filename = 'Library_Frontend.py'
    os.system(filename)
    os.system('notepad'+filename)

def __information__():
    filename = 'Std_info_FrontEnd.py'
    os.system(filename)
    os.system('notepad'+filename)

def __FeeReport__():
    filename = 'Fee_Frontend.py'
    os.system(filename)
    os.system('notepad'+filename)

def menu():
    root = Tk()
    root.title('Menu')
    root.geometry('1350x750')
    root.config(bg = 'navajo white')

    title_Frame = LabelFrame(root, font = ('arial',50,'bold'), width = 1000,
height = 100, bg = 'navajo white', relief = 'raise', bd = 13)
    title_Frame.grid(row = 0, column = 0, pady = 50)

    title_Label = Label(title_Frame, text = 'MENU', font = ('arial',30,'bold'),
bg = 'navajo white')
    title_Label.grid(row = 0, column = 0, padx = 150)

#=====FRAMES=====
Frame_1 = LabelFrame(root, font = ('arial',17,'bold'), width = 1000, height
= 100, bg = 'navajo white', relief = 'ridge', bd = 10)
Frame_1.grid(row = 1, column = 0, padx = 280)
Frame_2 = LabelFrame(root, font = ('arial',17,'bold'), width = 1000, height
= 100, bg = 'navajo white', relief = 'ridge', bd = 10)
Frame_2.grid(row = 2, column = 0, padx = 130, pady = 7)
Frame_3 = LabelFrame(root, font = ('arial',17,'bold'), width = 1000, height
= 100, bg = 'navajo white', relief = 'ridge', bd = 10)
Frame_3.grid(row = 3, column = 0, pady = 7)
Frame_4 = LabelFrame(root, font = ('arial',17,'bold'), width = 1000, height
= 100, bg = 'navajo white', relief = 'ridge', bd = 10)
Frame_4.grid(row = 4, column = 0, pady = 7)

#=====LABELS=====
Label_1 = Label(Frame_1, text = 'STUDENT PROFILE', font =
('arial',25,'bold'), bg = 'navajo white')
Label_1.grid(row = 0, column = 0, padx = 50, pady = 5)
```

```

        Label_2 = Label(Frame_2, text = 'FEE REPORT', font = ('arial',25,'bold'), bg
= 'navajo white')
        Label_2.grid(row = 0, column = 0, padx = 100, pady = 5)
        Label_3 = Label(Frame_3, text = 'LIBRARY SYSTEM', font =
('arial',25,'bold'), bg = 'navajo white')
        Label_3.grid(row = 0, column = 0, padx = 60, pady = 5)
        Label_4 = Label(Frame_4, text = 'MARKSHEET', font = ('arial',25,'bold'), bg
= 'navajo white')
        Label_4.grid(row = 0, column = 0, padx = 101, pady = 5)

#=====BUTTONS=====
=====
        Button_1 = Button(Frame_1, text = 'VIEW', font = ('arial',16,'bold'), width
= 8, command = __information__)
        Button_1.grid(row = 0, column = 3, padx = 50)
        Button_2 = Button(Frame_2, text = 'VIEW', font = ('arial',16,'bold'), width
= 8, command = __FeeReport__)
        Button_2.grid(row = 0, column = 3, padx = 50)
        Button_3 = Button(Frame_3, text = 'VIEW', font = ('arial',16,'bold'), width
= 8, command = __Library__)
        Button_3.grid(row = 0, column = 3, padx = 50)
        Button_4 = Button(Frame_4, text = 'VIEW', font = ('arial',16,'bold'), width
= 8, command = __marksheet__)
        Button_4.grid(row = 0, column = 3, padx = 50)

root.mainloop()

if __name__ == '__main__':
    menu()

```

2.student information

```

        try:
            global selected_tuple

            index = self.listbox.curselection()[0]
            selected_tuple = self.listbox.get(index)

            self.Entry_name.delete(0, END)
            self.Entry_name.insert(END, selected_tuple[1])
            self.Entry_fname.delete(0, END)
            self.Entry_fname.insert(END, selected_tuple[2])
            self.Entry_mname.delete(0, END)
            self.Entry_mname.insert(END,

selected_tuple[3])from tkinter import*
import tkinter.messagebox
import random
import Std_info_BackEnd
from tkinter import ttk

class Std_info():
    def __init__(self, master):
        self.master = master
        self.master.title('Student Information')
        self.master.geometry('1350x750')
        self.master.config(bg = 'navajowhite')

        def information():

#=====Variables=====

```

```

=====
        self.name = StringVar()
        self.fname = StringVar()
        self.mname = StringVar()
        self.address = StringVar()
        self.mobno = StringVar()
        self.email = StringVar()
        self.dob = StringVar()
        self.gender = StringVar()

#=====Functions=====
=====

        def StudentRec(event):
            self.Entry_address.delete(0, END)
            self.Entry_address.insert(END,
selected_tuple[4])

            self.Entry_mobno.delete(0, END)
            self.Entry_mobno.insert(END, selected_tuple[5])
            self.Entry_emailID.delete(0, END)
            self.Entry_emailID.insert(END,
selected_tuple[6])

            self.Entry_dob.delete(0, END)
            self.Entry_dob.insert(END, selected_tuple[7])
            self.Entry_gender.delete(0, END)
            self.Entry_gender.insert(END, selected_tuple[8])
        except IndexError:
            pass

        def Add():
            if(len(self.name.get()) != 0):
                Std_info_BackEnd.insert(self.name.get(),
self.fname.get(), self.mname.get(), self.address.get(), self.mobno.get(),
self.email.get(), self.dob.get(), \
                                self.gender.get())
                self.listbox.delete(0, END)
                self.listbox.insert(END, (self.name.get(),
self.fname.get(), self.mname.get(), self.address.get(), self.mobno.get(),
self.email.get(), self.dob.get(), \
                                self.gender.get()))

        def Display():
            self.listbox.delete(0, END)
            for row in Std_info_BackEnd.view():
                self.listbox.insert(END, row, str(' '))

        def Exit():
            Exit = tkinter.messagebox.askyesno("Login System",
"Confirm if you want to Exit")
            if Exit > 0:
                self.master.destroy()
                return

        def Reset():
            self.name.set('')
            self.fname.set('')
            self.mname.set('')
            self.address.set('')
            self.mobno.set('')
            self.email.set('')
            self.dob.set('')
            self.gender.set('')
            self.listbox.delete(0, END)

```

```

def Delete():
    if(len(self.name.get()) != 0):
        Std_info_BackEnd.delete(selected_tuple[0])
        Reset()
        Display()

def Search():
    self.listbox.delete(0, END)
    for row in Std_info_BackEnd.search(self.name.get(),
self.fname.get(), self.mname.get(), self.address.get(), self.mobno.get(),
self.email.get(), self.dob.get(),self.gender.get()):
        self.listbox.insert(END, row, str(' '))

def Update():
    if(len(self.name.get()) != 0):
        Std_info_BackEnd.delete(selected_tuple[0])
    if(len(self.name.get()) != 0):
        Std_info_BackEnd.insert(self.name.get(),
self.fname.get(), self.mname.get(), self.address.get(), self.mobno.get(),
self.email.get(), self.dob.get(), \
                                self.gender.get())

        self.listbox.delete(0, END)
        self.listbox.insert(END, (self.name.get(),
self.fname.get(), self.mname.get(), self.address.get(), self.mobno.get(),
self.email.get(), self.dob.get(), \
                                self.gender.get()))

#=====Frames=====
=====

self.Main_Frame = LabelFrame(self.master, width = 1300, height
= 500, font = ('arial',20,'bold'), \
                                bg = 'navajowhite',bd = 15,
relief = 'ridge')
self.Main_Frame.grid(row = 0, column = 0, padx = 10, pady =
20)

self.Frame_1 = LabelFrame(self.Main_Frame, width = 600, height
= 400, font = ('arial',15,'bold'), \
                                relief = 'ridge', bd = 10, bg =
'navajowhite', text = 'STUDENT INFORMATION ')
self.Frame_1.grid(row = 1, column = 0, padx = 10)

self.Frame_2 = LabelFrame(self.Main_Frame, width = 750, height
= 400, font = ('arial',15,'bold'), \
                                relief = 'ridge', bd = 10, bg =
'navajowhite', text = 'STUDENT DATABASE')
self.Frame_2.grid(row = 1, column = 1, padx = 5)

self.Frame_3 = LabelFrame(self.master, width = 1200, height =
100, font = ('arial',10,'bold'), \
                                bg = 'navajowhite', relief =
'ridge', bd = 13)
self.Frame_3.grid(row = 2, column = 0, pady = 10)

#=====Labels of
Frame_1=====
self.Label_name = Label(self.Frame_1, text = 'Name', font =

```

```

('arial',20,'bold'), bg = 'navajowhite')
self.Label_name.grid(row = 0, column = 0, sticky = W, padx =
20, pady = 10)
self.Label_fname = Label(self.Frame_1, text = 'Father Name',
font = ('arial',20,'bold'), bg = 'navajowhite')
self.Label_fname.grid(row = 1, column = 0, sticky = W, padx =
20)
self.Label_mname = Label(self.Frame_1, text = 'Mother Name',
font = ('arial',20,'bold'), bg = 'navajowhite')
self.Label_mname.grid(row = 2, column = 0, sticky = W, padx =
20)
self.Label_address = Label(self.Frame_1, text = 'Address',
font = ('arial',20,'bold'), bg = 'navajowhite')
self.Label_address.grid(row = 3, column = 0, sticky = W, padx
= 20)
self.Label_mobno = Label(self.Frame_1, text = 'Mobile Number',
font = ('arial',20,'bold'), bg = 'navajowhite')
self.Label_mobno.grid(row = 4, column = 0, sticky = W, padx =
20)
self.Label_emailID = Label(self.Frame_1, text = 'Email ID',
font = ('arial',20,'bold'), bg = 'navajowhite')
self.Label_emailID.grid(row = 5, column = 0, sticky = W, padx
= 20)
self.Label_dob = Label(self.Frame_1, text = 'Date of Birth',
font = ('arial',20,'bold'), bg = 'navajowhite')
self.Label_dob.grid(row = 6, column = 0, sticky = W, padx =
20)
self.Label_gender = Label(self.Frame_1, text = 'Gender', font
= ('arial',20,'bold'), bg = 'navajowhite')
self.Label_gender.grid(row = 7, column = 0, sticky = W, padx =
20, pady = 10)

#=====Entries of
Frame_1=====
self.Entry_name = Entry(self.Frame_1, font =
('arial',17,'bold'), textvariable = self.name)
self.Entry_name.grid(row = 0, column = 1, padx = 10, pady = 5)
self.Entry_fname = Entry(self.Frame_1, font =
('arial',17,'bold'), textvariable = self.fname)
self.Entry_fname.grid(row = 1, column = 1, padx = 10, pady =
5)
self.Entry_mname = Entry(self.Frame_1, font =
('arial',17,'bold'), textvariable = self.mname)
self.Entry_mname.grid(row = 2, column = 1, padx = 10, pady =
5)
self.Entry_address = Entry(self.Frame_1, font =
('arial',17,'bold'), textvariable = self.address)
self.Entry_address.grid(row = 3, column = 1, padx = 10, pady =
5)
self.Entry_mobno = Entry(self.Frame_1, font =
('arial',17,'bold'), textvariable = self.mobno)
self.Entry_mobno.grid(row = 4, column = 1, padx = 10, pady =
5)
self.Entry_emailID = Entry(self.Frame_1, font =
('arial',17,'bold'), textvariable = self.email)
self.Entry_emailID.grid(row = 5, column = 1, padx = 10, pady =
5)
self.Entry_dob = Entry(self.Frame_1, font =
('arial',17,'bold'), textvariable = self.dob)
self.Entry_dob.grid(row = 6, column = 1, padx = 10, pady = 5)
self.Entry_gender = ttk.Combobox(self.Frame_1, values = ('
', 'Male', 'Female', 'Others'),\
font = ('arial',17,'bold'),
textvariable = self.gender, width = 19)
self.Entry_gender.grid(row = 7, column = 1, padx = 10, pady =
5)

```

```

#=====Buttons of
self.Frame_3=====
        self.btnSave = Button(self.Frame_3, text = 'SAVE', font =
('arial',17,'bold'), width = 8, command = Add)
        self.btnSave.grid(row = 0, column = 0, padx = 10, pady = 10)
        self.btnDisplay = Button(self.Frame_3, text = 'DISPLAY', font
= ('arial',17,'bold'), width = 8, command = Display)
        self.btnDisplay.grid(row = 0, column = 1, padx = 10, pady =
10)
        self.btnReset = Button(self.Frame_3, text = 'RESET', font =
('arial',17,'bold'), width = 8, command = Reset)
        self.btnReset.grid(row = 0, column = 2, padx = 10, pady = 10)
        self.btnUpdate = Button(self.Frame_3, text = 'UPDATE', font =
('arial',17,'bold'), width = 8, command = Update)
        self.btnUpdate.grid(row = 0, column = 3, padx = 10, pady = 10)
        self.btnDelete = Button(self.Frame_3, text = 'DELETE', font =
('arial',17,'bold'), width = 8, command = Delete)
        self.btnDelete.grid(row = 0, column = 4, padx = 10, pady = 10)
        self.btnSearch = Button(self.Frame_3, text = 'SEARCH', font =
('arial',17,'bold'), width = 8, command = Search )
        self.btnSearch.grid(row = 0, column = 5, padx = 10, pady = 10)
        self.btnExit = Button(self.Frame_3, text = 'EXIT', font =
('arial',17,'bold'), width = 8, command = Exit)
        self.btnExit.grid(row = 0, column = 6, padx = 10, pady = 10)

#=====List Box and
self.scrollbar=====
        self.scrollbar = Scrollbar(self.Frame_2)
        self.scrollbar.grid(row = 0, column = 1, sticky = 'ns')

        self.listbox = Listbox(self.Frame_2, width = 75, height = 20 ,
font = ('arial',12,'bold'))
        self.listbox.bind('<<ListboxSelect>>', StudentRec)
        self.listbox.grid(row = 0, column = 0)
        self.scrollbar.config(command = self.listbox.yview)

        information()

root = Tk()
obj = Std_info(root)
root.mainloop()

```

3.Fee Report

```

from tkinter import*
from tkinter import ttk
import tkinter.messagebox
import datetime
import Fee_Backend

class Fee():
    def __init__(self, master):
        self.master = master
        self.master.title('Fee Report')
        self.master.geometry('1350x750')
        self.master.config(bg='Navajo white')

#
=====Variables=====
        self.recpt = StringVar()

```

```

self.name = StringVar()
self.admsn = StringVar()
self.date = StringVar()
self.branch = StringVar()
self.sem = StringVar()
self.total = DoubleVar()
self.paid = DoubleVar()
self.due = DoubleVar()

#
=====Functions=====
=====
def Tuple(event):
    try:
        global st
        index = self.list.curselection()[0]
        st = self.list.get(index)

        self.recpt_entry.delete(0, END)
        self.recpt_entry.insert(END, st[1])
        self.name_entry.delete(0, END)
        self.name_entry.insert(END, st[2])
        self.admsn_entry.delete(0, END)
        self.admsn_entry.insert(END, st[3])
        self.Date_entry.delete(0, END)
        self.Date_entry.insert(END, st[4])
        self.branch_entry.delete(0, END)
        self.branch_entry.insert(END, st[5])
        self.sem_entry.delete(0, END)
        self.sem_entry.insert(END, st[6])
        self.total_entry.delete(0, END)
        self.total_entry.insert(END, st[7])
        self.paid_entry.delete(0, END)
        self.paid_entry.insert(END, st[8])
        self.due_entry.delete(0, END)
        self.due_entry.insert(END, st[9])
    except IndexError:
        pass

def Insert():
    if (len(self.admsn.get()) != 0):
        Fee_Backend.insert(self.recpt.get(), self.name.get(),
self.admsn.get(), self.date.get(),
                                self.branch.get(), self.sem.get(),
self.total.get(), self.paid.get(),
                                self.due.get())
        self.list.delete(0, END)
        self.list.insert(END, (self.recpt.get(), self.name.get(),
self.admsn.get(), self.date.get(),
                                self.branch.get(), self.sem.get(),
self.total.get(), self.paid.get(),
                                self.due.get()))

def View():
    self.list.delete(0, END)
    for row in Fee_Backend.view():
        self.list.insert(END, row, str(' '))

def Reset():
    self.recpt.set(' ')
    self.name.set(' ')
    self.admsn.set(' ')
    #self.date.set(' ')
    self.branch.set(' ')
    self.sem.set(' ')
    self.paid.set(' ')
    self.due.set(' ')
    self.Display.delete('1.0', END)
    self.list.delete(0, END)

```

```
def Delete():
    Fee_Backend.delete(st[0])
    Reset()
    View()

def Receipt():
    self.Display.delete('1.0', END)
    self.Display.insert(END, '\t\tRECEIPT' + '\n\n')
    self.Display.insert(
        END, '\tReceipt No.\t      : ' + self.recpt.get() + '\n')
    self.Display.insert(END, '\tStudent Name   : ' +
                        self.name.get() + '\n')
    self.Display.insert(END, '\tAdmission No.\t: ' +
                        self.admsn.get() + '\n')
    self.Display.insert(
        END, '\tDate\t          : ' + self.date.get() + '\n')
    self.Display.insert(
        END, '\tBranch\t       : ' + self.branch.get() + '\n')
    self.Display.insert(
        END, '\tSemester \t     : ' + self.sem.get() + '\n\n')

    x1 = (self.var_1.get())
    x2 = (self.paid.get())
    x3 = (x1 - x2)

    self.Display.insert(END, '\tTotal Amount   : ' + str(x1) + '\n')
    self.Display.insert(END, '\tPaid Amount    : ' + str(x2) + '\n')
    self.Display.insert(END, '\tBalance\t      : ' + str(x3) + '\n')

    self.due.set(x3)

def Search():
    self.list.delete(0, END)
    for row in Fee_Backend.search(self.recpt.get(), self.name.get(),
self.admsn.get(), self.date.get(),
                                self.branch.get(), self.sem.get(),
self.total.get(), self.paid.get(),
                                self.due.get()):
        self.list.insert(END, row, str(' '))

def Update():
    Fee_Backend.delete(st[0])
    Insert()

def Exit():
    Exit = tkinter.messagebox.askyesno(
        'Attention', 'Confirm, if you want to Exit')
    if Exit > 0:
        root.destroy()
        return

#
=====Frames=====
=====
Main_Frame = Frame(self.master, bg='Navajo white')
Main_Frame.grid()

Title_Frame = LabelFrame(
    Main_Frame, width=1350, height=100, bg='Navajo white', relief='ridge',
bd=15)
Title_Frame.pack(side=TOP)

self.lblTitle = Label(Title_Frame, font=('arial', 40, 'bold'), text='FEE
REPORT',
                      bg='navajowhite', padx=13)
self.lblTitle.grid(padx=400)

Data_Frame = Frame(Main_Frame, width=1350, height=350,
```

```

        bg='Navajo white', relief='ridge', bd=15)
Data_Frame.pack(side=TOP, padx=15)

Frame_1 = LabelFrame(Data_Frame, width=850, height=350, bg='Navajo white',
relief='ridge', bd=8,
                        text='Informations', font=('arial', 15, 'bold'))
Frame_1.pack(side=LEFT, padx=10)

Frame_2 = LabelFrame(Data_Frame, width=495, height=350, bg='Navajo white',
relief='ridge', bd=8,
                        text='Fee Receipt', font=('arial', 15, 'bold'))
Frame_2.pack(side=RIGHT, padx=10)

List_Frame = Frame(Main_Frame, width=1350, height=150,
                    bg='Navajo white', relief='ridge', bd=15)
List_Frame.pack(side=TOP, padx=15)

Button_Frame = Frame(Main_Frame, width=1350, height=80,
                      bg='Navajo white', relief='ridge', bd=15)
Button_Frame.pack(side=TOP)

#
=====Labels=====
=====
self.recpt_label = Label(Frame_1, text='Receipt No. : ', font=(
    'arial', 14, 'bold'), bg='Navajo white')
self.recpt_label.grid(row=0, column=0, padx=15, sticky=W)

self.name_label = Label(Frame_1, text='Student Name : ', font=(
    'arial', 14, 'bold'), bg='Navajo white')
self.name_label.grid(row=1, column=0, padx=15, sticky=W)

self.admsn_label = Label(Frame_1, text='Admission No. : ', font=(
    'arial', 14, 'bold'), bg='Navajo white')
self.admsn_label.grid(row=2, column=0, padx=15, sticky=W)

self.Date_label = Label(Frame_1, text='Date : ', font=(
    'arial', 14, 'bold'), bg='Navajo white')
self.Date_label.grid(row=3, column=0, padx=15, sticky=W)

self.branch_label = Label(Frame_1, text='Branch : ', font=(
    'arial', 14, 'bold'), bg='Navajo white')
self.branch_label.grid(row=4, column=0, padx=15, sticky=W)

self.sem_label = Label(Frame_1, text='Semester : ', font=(
    'arial', 14, 'bold'), bg='Navajo white')
self.sem_label.grid(row=5, column=0, padx=15, sticky=W)

self.total_label = Label(Frame_1, text='TOTAL AMOUNT : ', font=(
    'arial', 14, 'bold'), bg='Navajo white')
self.total_label.grid(row=2, column=2, padx=5, sticky=W)

self.paid_label = Label(Frame_1, text='PAID AMOUNT : ', font=(
    'arial', 14, 'bold'), bg='Navajo white')
self.paid_label.grid(row=3, column=2, padx=5, sticky=W)

self.due_label = Label(Frame_1, text='BALANCE : ', font=(
    'arial', 14, 'bold'), bg='Navajo white')
self.due_label.grid(row=4, column=2, padx=5, sticky=W)

#
=====Entries=====
=====
self.var_1 = DoubleVar(Frame_1, value='36265')
d1 = datetime.date.today()
self.date.set(d1)

self.recpt_entry = Entry(Frame_1, font=(
    'arial', 14), textvariable=self.recpt)

```

```

self.recpt_entry.grid(row=0, column=1, padx=15, pady=5)

self.name_entry = Entry(Frame_1, font=(
    'arial', 14), textvariable=self.name)
self.name_entry.grid(row=1, column=1, padx=15, pady=5)

self.admsn_entry = Entry(Frame_1, font=(
    'arial', 14), textvariable=self.admsn)
self.admsn_entry.grid(row=2, column=1, padx=15, pady=5)

self.Date_entry = Entry(Frame_1, font=(
    'arial', 14), textvariable=self.date)
self.Date_entry.grid(row=3, column=1, padx=15, pady=5)

self.branch_entry = ttk.Combobox(Frame_1, values=(' ', 'CSE', 'IT',
'ETC/ET&T', 'Mechanical', 'Civil', 'EE', 'EEE'),
                                font=('arial', 14), width=19,
textvariable=self.branch)
self.branch_entry.grid(row=4, column=1, padx=15, pady=5)

self.sem_entry = ttk.Combobox(Frame_1, values=(' ', 'FIRST', 'SECOND',
'THIRD', 'FOURTH', 'FIFTH', 'SIXTH',
                                'SEVENTH', 'EIGHTH'),
                                font=('arial', 14), width=19,
                                textvariable=self.sem)
self.sem_entry.grid(row=5, column=1, padx=15, pady=5)

self.total_entry = Entry(Frame_1, font=(
    'arial', 14), width=10, textvariable=self.var_1, state='readonly')
self.total_entry.grid(row=2, column=3, padx=8, pady=5)

self.paid_entry = Entry(Frame_1, font=(
    'arial', 14), width=10, textvariable=self.paid)
self.paid_entry.grid(row=3, column=3, pady=5)

self.due_entry = Entry(Frame_1, font=(
    'arial', 14), width=10, textvariable=self.due)
self.due_entry.grid(row=4, column=3, pady=7)

#
=====Frame_2=====
=====
self.Display = Text(Frame_2, width=42, height=12,
                    font=('arial', 14, 'bold'))
self.Display.grid(row=0, column=0, padx=3)

# =====List box and
scrollbar=====
sb = Scrollbar(List_Frame)
sb.grid(row=0, column=1, sticky='ns')

self.list = Listbox(List_Frame, font=(
    'arial', 13, 'bold'), width=140, height=8)
self.list.bind('<<ListboxSelect>>', Tuple)
self.list.grid(row=0, column=0)
sb.config(command=self.list.yview)

#
=====Buttons=====
=====
btnSave = Button(Button_Frame, text='SAVE', font=(
    'arial', 14, 'bold'), width=10, command=Insert)
btnSave.grid(row=0, column=0, padx=5, pady=5)

btnDisplay = Button(Button_Frame, text='DISPLAY', font=(
    'arial', 14, 'bold'), width=10, command=View)
btnDisplay.grid(row=0, column=1, padx=5, pady=5)

btnReset = Button(Button_Frame, text='RESET', font=(

```

```

        'arial', 14, 'bold'), width=10, command=Reset)
btnReset.grid(row=0, column=2, padx=5, pady=5)

btnReset = Button(Button_Frame, text='UPDATE', font=(
    'arial', 14, 'bold'), width=10, command=Update)
btnReset.grid(row=0, column=3, padx=5, pady=5)

btnSearch = Button(Button_Frame, text='SEARCH', font=(
    'arial', 14, 'bold'), width=10, command=Search)
btnSearch.grid(row=0, column=4, padx=5, pady=5)

btnDelete = Button(Button_Frame, text='DELETE', font=(
    'arial', 14, 'bold'), width=10, command=Delete)
btnDelete.grid(row=0, column=5, padx=5, pady=5)

btnReceipt = Button(Button_Frame, text='RECEIPT', font=(
    'arial', 14, 'bold'), width=10, command=Receipt)
btnReceipt.grid(row=0, column=6, padx=5, pady=5)

btnExit = Button(Button_Frame, text='EXIT', font=(
    'arial', 14, 'bold'), width=10, command=Exit)
btnExit.grid(row=0, column=7, padx=5, pady=5)

root = Tk()
obj = Fee(root)
root.mainloop()

```

4. Library System

```

from tkinter import*
from tkinter import ttk
import random
from datetime import datetime
import tkinter.messagebox
import Library_Backend

class Library:

    def __init__(self, root):
        self.root = root
        self.root.title('Library Management System')
        self.root.geometry('1350x750')
        self.root.config(bg = 'navajowhite')

#=====Variables=====
self.Mtype = StringVar()
self.refno = StringVar()
self.fname = StringVar()
self.surname = StringVar()
self.address = StringVar()
self.post = StringVar()
self.mobno = StringVar()
self.ID = StringVar()
self.title = StringVar()
self.author = StringVar()
self.borrow = StringVar()
self.due = StringVar()
self.loan = StringVar()
self.yop = StringVar()
self.edsn = StringVar()

#=====Functions=====

```

```

=====
def BookRec(event):
    try:
        global selected_tuple
        index = self.Listbox_2.curselection()[0]
        selected_tuple = self.Listbox_2.get(index)

        self.Entry_0.delete(0, END)
        self.Entry_0.insert(END, selected_tuple[1])
        self.Entry_1.delete(0, END)
        self.Entry_1.insert(END, selected_tuple[2])
        self.Entry_2.delete(0, END)
        self.Entry_2.insert(END, selected_tuple[3])
        self.Entry_3.delete(0, END)
        self.Entry_3.insert(END, selected_tuple[4])
        self.Entry_4.delete(0, END)
        self.Entry_4.insert(END, selected_tuple[5])
        self.Entry_5.delete(0, END)
        self.Entry_5.insert(END, selected_tuple[6])
        self.Entry_6.delete(0, END)
        self.Entry_6.insert(END, selected_tuple[7])
        self.Entry_7.delete(0, END)
        self.Entry_7.insert(END, selected_tuple[8])
        self.Entry_8.delete(0, END)
        self.Entry_8.insert(END, selected_tuple[9])
        self.Entry_9.delete(0, END)
        self.Entry_9.insert(END, selected_tuple[10])
        self.Entry_10.delete(0, END)
        self.Entry_10.insert(END, selected_tuple[11])
        self.Entry_11.delete(0, END)
        self.Entry_11.insert(END, selected_tuple[12])
        self.Entry_12.delete(0, END)
        self.Entry_12.insert(END, selected_tuple[13])

    except IndexError:
        pass

def Insert():
    if(len(self.refno.get()) != 0):
        Library_Backend.insert(self.Mtype.get(),
self.refno.get(), self.fname.get(), self.surname.get()\
, self.address.get(),
self.post.get(), self.mobno.get(), self.ID.get()\
, self.title.get(),
self.author.get(), self.borrow.get(), self.due.get()\
, self.loan.get())
        self.Listbox_2.delete(0, END)
        self.Listbox_2.insert(END, (self.Mtype.get(),
self.refno.get(), self.fname.get(), self.surname.get()\
, self.address.get(),
self.post.get(), self.mobno.get(), self.ID.get()\
, self.title.get(),
self.author.get(), self.borrow.get(), self.due.get()\
, self.loan.get()))

def Display():
    self.Listbox_2.delete(0, END)
    for row in Library_Backend.view():
        self.Listbox_2.insert(END, row, str(' '))

def Exit():
    Exit = tkinter.messagebox.askyesno('Library Management
System','Confirm if you want to Exit')
    if Exit > 0:
        root.destroy()
        return

def Reset():

```

```

        self.Mtype.set('')
        self.refno.set('')
        self.fname.set('')
        self.surname.set('')
        self.address.set('')
        self.post.set('')
        self.mobno.set('')
        self.ID.set('')
        self.title.set('')
        self.author.set('')
        self.borrow.set('')
        self.due.set('')
        self.loan.set('')
        self.Display.delete('1.0',END)
        self.Listbox_2.delete(0, END)

    def Delete():
        Library_Backend.delete(selected_tuple[0])
        Reset()
        Display()

    def Update():
        Library_Backend.delete(selected_tuple[0])
        Library_Backend.insert(self.Mtype.get(), self.refno.get(),
self.fname.get(), self.surname.get() \
                                , self.address.get(),
self.post.get(), self.mobno.get(), self.ID.get() \
                                , self.title.get(),
self.author.get(), self.borrow.get(), self.due.get() \
                                , self.loan.get())
        self.Listbox_2.delete(0, END)
        self.Listbox_2.insert(END, (self.Mtype.get(), self.refno.get(),
self.fname.get(), self.surname.get() \
                                , self.address.get(),
self.post.get(), self.mobno.get(), self.ID.get() \
                                , self.title.get(),
self.author.get(), self.borrow.get(), self.due.get() \
                                , self.loan.get()))

    def Search():
        self.Listbox_2.delete(0, END)
        for row in Library_Backend.search(self.Mtype.get(),
self.refno.get(), self.fname.get(), self.surname.get() \
                                , self.address.get(),
self.post.get(), self.mobno.get(), self.ID.get() \
                                , self.title.get(),
self.author.get(), self.borrow.get(), self.due.get() \
                                , self.loan.get()):
            self.Listbox_2.insert(END, row, str(' '))

    def Details():
        self.Display.delete('1.0',END)
        self.Display.insert(END, 'Book ID: ' + self.ID.get() + '\n')
        self.Display.insert(END, 'Title: ' + self.title.get() + '\n')
        self.Display.insert(END, 'Author: ' + self.author.get() +
'\n')

        self.Display.insert(END, 'Edition: ' + self.edsn.get() + '\n')
        self.Display.insert(END, 'Year of Publisison: \t' +
self.yop.get() + '\n')
        self.Display.insert(END, 'Date Borrowed: ' + self.borrow.get()
+ '\n')

        self.Display.insert(END, 'Date Due: ' + self.due.get() + '\n')
        self.Display.insert(END, 'Days in Loan: ' + self.loan.get() +
'\n')

```

```

#=====Frames=====
=====

```

```

Main_Frame = Frame(self.root, bg = 'navajowhite')
Main_Frame.grid()

Title_Frame_1 = Frame(Main_Frame, width = 1350, bg = 'navajowhite',
relief = RIDGE, bd = 15, padx = 20)
Title_Frame_1.pack(side = TOP)

self.lblTitle = Label(Title_Frame_1, font = ('arial',40,'bold'), text
= '\tLibrary Management System\t', \
                        bg = 'navajowhite', padx = 13)
self.lblTitle.grid()

Button_Frame = Frame(Main_Frame, width = 1350, height = 50, relief =
RIDGE, bd = 10, bg = 'navajowhite')
Button_Frame.pack(side = BOTTOM)

Detail_Frame = Frame(Main_Frame, width = 1350, height = 100, relief =
RIDGE, bd = 10, bg = 'navajowhite')
Detail_Frame.pack(side = BOTTOM)

Data_Frame = Frame(Main_Frame, width = 1350, height = 400, relief =
RIDGE, bd = 15, bg = 'navajowhite')
Data_Frame.pack(side = BOTTOM)

Frame_1 = LabelFrame(Data_Frame, width = 800, height = 400, relief =
RIDGE, bd = 10, bg = 'navajowhite', \
                      text = "Library Membership Info:", padx = 20, font =
('arial',15,'bold'))
Frame_1.pack(side = LEFT, padx = 3)

Frame_2 = LabelFrame(Data_Frame, width = 550, height = 400, relief =
RIDGE, bd = 10, bg = 'navajowhite', \
                      text = "Book Details:", padx = 20, font =
('arial',15,'bold'))
Frame_2.pack(side = RIGHT)

#=====Labels=====
=====
self.Label_1 = Label(Frame_1, text = 'Member type', font =
('arial',13,'bold'), pady = 2, \
                      bg = 'navajowhite' )
self.Label_1.grid(row = 0, column = 0, sticky = W)
self.Label_2 = Label(Frame_1, text = 'Reference No.', font =
('arial',13,'bold'), pady = 2, \
                      bg = 'navajowhite' )
self.Label_2.grid(row = 1, column = 0, sticky = W)
self.Label_3 = Label(Frame_1, text = 'First Name', font =
('arial',13,'bold'), pady = 2, \
                      bg = 'navajowhite' )
self.Label_3.grid(row = 2, column = 0, sticky = W)
self.Label_4 = Label(Frame_1, text = 'Surname', font =
('arial',13,'bold'), pady = 2, \
                      bg = 'navajowhite' )
self.Label_4.grid(row = 3, column = 0, sticky = W)
self.Label_5 = Label(Frame_1, text = 'Address', font =
('arial',13,'bold'), pady = 2, \
                      bg = 'navajowhite' )
self.Label_5.grid(row = 4, column = 0, sticky = W)
self.Label_6 = Label(Frame_1, text = 'Post Code', font =
('arial',13,'bold'), pady = 2, \
                      bg = 'navajowhite' )
self.Label_6.grid(row = 5, column = 0, sticky = W)
self.Label_7 = Label(Frame_1, text = 'Mobile No.', font =
('arial',13,'bold'), pady = 2, \
                      bg = 'navajowhite' )
self.Label_7.grid(row = 6, column = 0, sticky = W)
self.Label_8 = Label(Frame_1, text = 'Book ID', font =

```

```

('arial',13,'bold'), pady = 2, \
                                bg = 'navajowhite' )
    self.Label_8.grid(row = 0, column = 2, sticky = W)
    self.Label_9 = Label(Frame_1, text = 'Book Title', font =
('arial',13,'bold'), pady = 2, \
                                bg = 'navajowhite' )
    self.Label_9.grid(row = 1, column = 2, sticky = W)
    self.Label_10 = Label(Frame_1, text = 'Author', font =
('arial',13,'bold'), pady = 2, \
                                bg = 'navajowhite' )
    self.Label_10.grid(row = 2, column = 2, sticky = W)
    self.Label_11 = Label(Frame_1, text = 'Date Borrowed', font =
('arial',13,'bold'), pady = 2, \
                                bg = 'navajowhite' )
    self.Label_11.grid(row = 3, column = 2, sticky = W)
    self.Label_13 = Label(Frame_1, text = 'Date Due', font =
('arial',13,'bold'), pady = 2, \
                                bg = 'navajowhite' )
    self.Label_13.grid(row = 4, column = 2, sticky = W)
    self.Label_13 = Label(Frame_1, text = 'Days in Loan', font =
('arial',13,'bold'), pady = 2, \
                                bg = 'navajowhite' )
    self.Label_13.grid(row = 5, column = 2, sticky = W)

#=====Entries=====
=====
    self.Entry_0 = ttk.Combobox(Frame_1, values = (
', 'Student', 'Faculty', 'Staff Member'), \
                                font = ('arial',13,'bold'), width = 23,
textvariable = self.Mtype )
    self.Entry_0.grid(row = 0, column = 1)
    self.Entry_1 = Entry(Frame_1, font = ('arial',13,'bold'), width = 25,
textvariable = self.refno )
    self.Entry_1.grid(row = 1, column = 1, padx = 15)
    self.Entry_2 = Entry(Frame_1, font = ('arial',13,'bold'), width = 25,
textvariable = self.fname)
    self.Entry_2.grid(row = 2, column = 1, padx = 15)
    self.Entry_3 = Entry(Frame_1, font = ('arial',13,'bold'), width = 25,
textvariable = self.surname)
    self.Entry_3.grid(row = 3, column = 1, padx = 15)
    self.Entry_4 = Entry(Frame_1, font = ('arial',13,'bold'), width = 25,
textvariable = self.address)
    self.Entry_4.grid(row = 4, column = 1, padx = 15)
    self.Entry_5 = Entry(Frame_1, font = ('arial',13,'bold'), width = 25,
textvariable = self.post)
    self.Entry_5.grid(row = 5, column = 1, padx = 15)
    self.Entry_6 = Entry(Frame_1, font = ('arial',13,'bold'), width = 25,
textvariable = self.mobno)
    self.Entry_6.grid(row = 6, column = 1, padx = 15)
    self.Entry_7 = Entry(Frame_1, font = ('arial',13,'bold'), width = 25,
textvariable = self.ID)
    self.Entry_7.grid(row = 0, column = 4, padx = 15)
    self.Entry_8 = Entry(Frame_1, font = ('arial',13,'bold'), width = 25,
textvariable = self.title)
    self.Entry_8.grid(row = 1, column = 4, padx = 15)
    self.Entry_9 = Entry(Frame_1, font = ('arial',13,'bold'), width = 25,
textvariable = self.author)
    self.Entry_9.grid(row = 2, column = 4, padx = 15)
    self.Entry_10 = Entry(Frame_1, font = ('arial',13,'bold'), width =
25, textvariable = self.borrow)
    self.Entry_10.grid(row = 3, column = 4, padx = 15)
    self.Entry_11 = Entry(Frame_1, font = ('arial',13,'bold'), width =
25, textvariable = self.due)
    self.Entry_11.grid(row = 4, column = 4, padx = 15)
    self.Entry_12 = Entry(Frame_1, font = ('arial',13,'bold'), width =
25, textvariable = self.loan)

```

```

self.Entry_12.grid(row = 5, column = 4, padx = 15)

#=====Widgets=====
self.Display = Text(Frame_2, font = ('arial',13,'bold'), width = 28,
height = 11)
self.Display.grid(row = 0, column = 2)

List_of_Books = [' C ',' C++',' Java',' Python',' PHP',' Java
Script',' My SQL',' Data Structure',' Linux',' \
                ' Operating System',' Web Developement',' Data
Science',' Algorithms',' Android',' \
                ' VB.net']

#=====Function for Books
Details=====
def SelectedBook(event):
    value = str(self.Listbox_1.get(self.Listbox_1.curselection()))
    v = value

    if (v == ' C '):
        self.ID.set('ISBN 525341')
        self.title.set('Programming using C')
        self.author.set('Yashwant Kanetkar')
        self.yop.set('2019')
        self.edsn.set('5th')

        import datetime

        d1 = datetime.date.today()
        d2 = datetime.timedelta(days = 14)
        d3 = (d1 + d2)
        self.borrow.set(d1)
        self.loan.set('14')
        self.due.set(d3)
        Details()
    elif (v == ' C++'):
        self.ID.set('ISBN 345687')
        self.title.set('Programming using C++')
        self.author.set('Yashwant Kanetkar')
        self.yop.set('2019')
        self.edsn.set('4th')

        import datetime

        d1 = datetime.date.today()
        d2 = datetime.timedelta(days = 10)
        d3 = (d1 + d2)
        self.borrow.set(d1)
        self.loan.set('10')
        self.due.set(d3)
        Details()
    elif (v == ' Java'):
        self.ID.set('ISBN 643842')
        self.title.set('Java Programming')
        self.author.set('Joshua Bloch')
        self.yop.set('2019')
        self.edsn.set('7th')

        import datetime

        d1 = datetime.date.today()
        d2 = datetime.timedelta(days = 13)
        d3 = (d1 + d2)
        self.borrow.set(d1)

```

```

        self.loan.set('13')
        self.due.set(d3)
        Details()
elif (v == ' Python'):
    self.ID.set('ISBN 564524')
    self.title.set('Python Programming')
    self.author.set('John Zelle')
    self.yop.set('2019')
    self.edsn.set('3rd')

    import datetime

    d1 = datetime.date.today()
    d2 = datetime.timedelta(days = 13)
    d3 = (d1 + d2)
    self.borrow.set(d1)
    self.loan.set('13')
    self.due.set(d3)
    Details()
elif (v == ' PHP'):
    self.ID.set('ISBN 735893')
    self.title.set('PHP Programming')
    self.author.set('Alan Forbes')
    self.yop.set('2019')
    self.edsn.set('5th')

    import datetime

    d1 = datetime.date.today()
    d2 = datetime.timedelta(days = 15)
    d3 = (d1 + d2)
    self.borrow.set(d1)
    self.loan.set('15')
    self.due.set(d3)
    Details()
elif (v == ' Java Script'):
    self.ID.set('ISBN 643842')
    self.title.set('Java Script Programming')
    self.author.set('Jon Duckett.')
    self.yop.set('2019')
    self.edsn.set('4th')

    import datetime

    d1 = datetime.date.today()
    d2 = datetime.timedelta(days = 13)
    d3 = (d1 + d2)
    self.borrow.set(d1)
    self.loan.set('13')
    self.due.set(d3)
    Details()
elif (v == ' My SQL'):
    self.ID.set('ISBN 649635')
    self.title.set('My SQL Programming')
    self.author.set('Groff James')
    self.yop.set('2019')
    self.edsn.set('3rd')

    import datetime

    d1 = datetime.date.today()
    d2 = datetime.timedelta(days = 20)
    d3 = (d1 + d2)
    self.borrow.set(d1)
    self.loan.set('20')
    self.due.set(d3)
    Details()
elif (v == ' Data Structure'):
    self.ID.set('ISBN 531588')

```

```

        self.title.set('Data Structure')
        self.author.set('Karumanchi Narasimha')
        self.yop.set('2019')
        self.edsn.set('5th')

        import datetime

        d1 = datetime.date.today()
        d2 = datetime.timedelta(days = 11)
        d3 = (d1 + d2)
        self.borrow.set(d1)
        self.loan.set('11')
        self.due.set(d3)
        Details()
    elif (v == 'Linux'):
        self.ID.set('ISBN 356853')
        self.title.set('Linux Administration')
        self.author.set('SOYINKA')
        self.yop.set('2019')
        self.edsn.set('1st')

        import datetime

        d1 = datetime.date.today()
        d2 = datetime.timedelta(days = 6)
        d3 = (d1 + d2)
        self.borrow.set(d1)
        self.loan.set('6')
        self.due.set(d3)
        Details()
    elif (v == 'Operating System'):
        self.ID.set('ISBN 536453')
        self.title.set('OS Concepts ')
        self.author.set('Silberschatz Abraham')
        self.yop.set('2019')
        self.edsn.set('4th')

        import datetime

        d1 = datetime.date.today()
        d2 = datetime.timedelta(days = 12)
        d3 = (d1 + d2)
        self.borrow.set(d1)
        self.loan.set('12')
        self.due.set(d3)
        Details()
    elif (v == 'Web Development'):
        self.ID.set('ISBN 543548')
        self.title.set('Web Developement ')
        self.author.set('Paul McFedries')
        self.yop.set('2019')
        self.edsn.set('3rd')

        import datetime

        d1 = datetime.date.today()
        d2 = datetime.timedelta(days = 15)
        d3 = (d1 + d2)
        self.borrow.set(d1)
        self.loan.set('15')
        self.due.set(d3)
        Details()
    elif (v == 'Data Science'):
        self.ID.set('ISBN 835764')
        self.title.set('Data Science Concept ')
        self.author.set('David Stephenson')
        self.yop.set('2019')
        self.edsn.set('3rd')

```

```

import datetime

d1 = datetime.date.today()
d2 = datetime.timedelta(days = 15)
d3 = (d1 + d2)
self.borrow.set(d1)
self.loan.set('15')
self.due.set(d3)
Details()
elif (v == ' Algorithms'):
    self.ID.set('ISBN 535674')
    self.title.set('Basics of Algorithm ')
    self.author.set('Karumanchi Narasimha')
    self.yop.set('2019')
    self.edsn.set('7th')

import datetime

d1 = datetime.date.today()
d2 = datetime.timedelta(days = 10)
d3 = (d1 + d2)
self.borrow.set(d1)
self.loan.set('10')
self.due.set(d3)
Details()
elif (v == ' Android'):
    self.ID.set('ISBN 356452')
    self.title.set('Android Programming')
    self.author.set('Harwani B. M')
    self.yop.set('2019')
    self.edsn.set('4th')

import datetime

d1 = datetime.date.today()
d2 = datetime.timedelta(days = 9)
d3 = (d1 + d2)
self.borrow.set(d1)
self.loan.set('9')
self.due.set(d3)
Details()

#=====List Box and Scroll
Bar=====
sb_1 = Scrollbar(Frame_2)
sb_1.grid(row =0, column = 1, sticky = 'ns')

self.Listbox_1 = Listbox(Frame_2, font = ('arial',13,'bold'), width =
20, height = 10)
self.Listbox_1.bind('<<ListboxSelect>>', SelectedBook)
self.Listbox_1.grid(row = 0, column = 0)
sb_1.config(command = self.Listbox_1.yview)

sb_2 = Scrollbar(Detail_Frame)
sb_2.grid(row = 1, column = 1, sticky = 'ns')

self.Listbox_2 = Listbox(Detail_Frame, font = ('arial',13,'bold'),
width = 144, height = 11)
self.Listbox_2.bind('<<ListboxSelect>>', BookRec)
self.Listbox_2.grid(row = 1, column = 0)
sb_2.config(command = self.Listbox_2.yview)

for items in List_of_Books:
    self.Listbox_1.insert(END, items)

#=====Buttons=====

```

```

=====
        Button_1 = Button(Button_Frame, text = 'SAVE', font =
('arial',15,'bold'), width = 10, command = Insert)
        Button_1.grid(row = 0, column = 0, padx = 8, pady = 5)
        Button_2 = Button(Button_Frame, text = 'DISPLAY', font =
('arial',15,'bold'), width = 10, command = Display)
        Button_2.grid(row = 0, column = 1, padx = 8)
        Button_3 = Button(Button_Frame, text = 'RESET', font =
('arial',15,'bold'), width = 10, command = Reset)
        Button_3.grid(row = 0, column = 2, padx = 8)
        Button_4 = Button(Button_Frame, text = 'UPDATE', font =
('arial',15,'bold'), width = 10, command = Update)
        Button_4.grid(row = 0, column = 3, padx = 8)
        Button_5 = Button(Button_Frame, text = 'SEARCH', font =
('arial',15,'bold'), width = 10, command = Search)
        Button_5.grid(row = 0, column = 4, padx = 8)
        Button_6 = Button(Button_Frame, text = 'DELETE', font =
('arial',15,'bold'), width = 10, command = Delete)
        Button_6.grid(row = 0, column = 5, padx = 8)
        Button_7 = Button(Button_Frame, text = 'EXIT', font =
('arial',15,'bold'), width = 10, command = Exit)
        Button_7.grid(row = 0, column = 6, padx = 8)

if __name__ == '__main__':
    root = Tk()
    application = Library(root)
    root.mainloop()

```

5. Marksheet

```

from tkinter import *
import random
import Marksheet_Backend
import tkinter.messagebox
from tkinter import ttk

def marksheet():
    root = Tk()
    root.title('Marksheet')
    root.geometry('1350x750')
    root.config(bg = 'Navajo white')

#=====Variables=====
=====
    name = StringVar()
    roll = StringVar()
    fname = StringVar()
    mname = StringVar()
    DOB = StringVar()
    gender = StringVar()
    scl = StringVar()
    email = StringVar()
    m1 = DoubleVar()
    m2 = DoubleVar()
    m3 = DoubleVar()
    m4 = DoubleVar()
    m5 = DoubleVar()
    gt = DoubleVar()
    per = DoubleVar()
    cgpa = DoubleVar()
    grade = StringVar()
    div = StringVar()
    result = StringVar()

```

```

#=====Functions=====
=====
def Add():
    if (len(roll.get()) != 0):

Marksheet_Backend.insert(name.get(), roll.get(), fname.get(), mname.get(), DOB.get(), ge
nder.get(), \

scl.get(), email.get(), m1.get(), m2.get(), m3.get(), m4.get(), m5.get(), \

gt.get(), per.get(), cgpa.get(), grade.get(), div.get(), result.get())

def Update():
    if (len(roll.get()) != 0):

Marksheet_Backend.update(name.get(), roll.get(), fname.get(), mname.get(), DOB.get(), ge
nder.get(), \

scl.get(), email.get(), m1.get(), m2.get(), m3.get(), m4.get(), m5.get(), \

gt.get(), per.get(), cgpa.get(), grade.get(), div.get(), result.get())

def Exit():
    Exit = tkinter.messagebox.askyesno('Marksheet', 'Confirm if you want
to Exit')
    if Exit > 0:
        root.destroy()
        return

def Compute():
    x1 = (m1.get());      x2 = (m2.get());      x3 = (m3.get());      x4 =
(m4.get());      x5 = (m5.get())

    if x1 > 100:
        tkinter.messagebox.askokcancel('Attention', 'Please enter
Correct Marks')
        return
    if x2 > 100:
        tkinter.messagebox.askokcancel('Attention', 'Please enter
Correct Marks')
        return
    if x3 > 100:
        tkinter.messagebox.askokcancel('Attention', 'Please enter
Correct Marks')
        return
    if x4 > 100:
        tkinter.messagebox.askokcancel('Attention', 'Please enter
Correct Marks')
        return
    if x5 > 100:
        tkinter.messagebox.askokcancel('Attention', 'Please enter
Correct Marks')
        return

    tot = x1+x2+x3+x4+x5
    gt.set(tot)

    Per = ((x1+x2+x3+x4+x5) * 100)/500
    per.set(Per)

    cg = (((x1+x2+x3+x4+x5) * 100)/500) / 9.5
    cgpa.set(round(cg,1))

    if cg > 10:
        cgpa.set(10)

```

```

if ((x1+x2+x3+x4+x5) * 100)/500) <= 40:
    grd = 'G'
elif ((x1+x2+x3+x4+x5) * 100)/500) <= 50:
    grd = 'F'
elif ((x1+x2+x3+x4+x5) * 100)/500) <= 60:
    grd = 'E'
elif ((x1+x2+x3+x4+x5) * 100)/500) <= 70:
    grd = 'D'
elif ((x1+x2+x3+x4+x5) * 100)/500) <= 80:
    grd = 'C'
elif ((x1+x2+x3+x4+x5) * 100)/500) <= 90:
    grd = 'B'
else:
    grd = 'A'

grade.set(grd)

count = 0
if x1 < 33:
    count = count + 1
if x2 < 33:
    count = count + 1
if x3 < 33:
    count = count + 1
if x4 < 33:
    count = count + 1
if x5 < 33:
    count = count + 1

if (count == 0):
    result.set('PASS')
elif (count == 1 or count == 2 ):
    result.set('SUPPLY')
else:
    result.set('FAIL')

if Per <= 45 and result != "FAIL":
    div.set('THIRD')
elif Per <= 60 and result != "FAIL":
    div.set('SECOND')
elif Per <= 100:
    div.set('FIRST')

def Reset():
    name.set(' ')
    roll.set(' ')
    fname.set(' ')
    mname.set(' ')
    DOB.set(' ')
    gender.set(' ')
    scl.set(' ')
    email.set(' ')
    m1.set(' ')
    m2.set(' ')
    m3.set(' ')
    m4.set(' ')
    m5.set(' ')
    gt.set(' ')
    per.set(' ')
    cgpa.set(' ')
    grade.set(' ')
    div.set(' ')
    result.set(' ')

```

```

Frame_1 = LabelFrame(root, width = 1200, height = 400, font =
('arial',20,'bold'), bg = 'Navajo white', bd = 10, \
    text = 'Student Details', relief = 'ridge')
Frame_1.grid(row = 1, column = 0, pady = 20, padx = 20)

#=====Labels and Entries for
Frame_1=====
Label_Name = Label(Frame_1, text = 'Name', font = ('arial',15,'bold'), bg =
'Navajo white')
Label_Name.grid(row = 0, column = 0, padx = 80)
Entry_Name = Entry(Frame_1, font = ('arial',15), width = 25, textvariable =
name)
Entry_Name.grid(row = 0, column = 1, padx = 5, pady = 5)

Label_Roll_no = Label(Frame_1, text = 'Roll Number', font =
('arial',15,'bold'), bg = 'Navajo white')
Label_Roll_no.grid(row = 0, column = 3, padx = 80)
Entry_Roll_no = Entry(Frame_1, font = ('arial',15), width = 25, textvariable
= roll)
Entry_Roll_no.grid(row = 0, column = 4, padx = 40)

Label_Father_Name = Label(Frame_1, text = 'Father Name', font =
('arial',15,'bold'), bg = 'Navajo white')
Label_Father_Name.grid(row = 1, column = 0, padx = 80)
Entry_Father_Name = Entry(Frame_1, font = ('arial',15), width = 25,
textvariable = fname)
Entry_Father_Name.grid(row = 1, column = 1, padx = 5, pady = 10)

Label_Mother_Name = Label(Frame_1, text = 'Mother Name', font =
('arial',15,'bold'), bg = 'Navajo white')
Label_Mother_Name.grid(row = 1, column = 3, padx = 80)
Entry_Mother_Name = Entry(Frame_1, font = ('arial',15), width = 25,
textvariable = mname)
Entry_Mother_Name.grid(row = 1, column = 4, padx = 5)

Label_DOB = Label(Frame_1, text = 'Date of Birth', font =
('arial',15,'bold'), bg = 'Navajo white')
Label_DOB.grid(row = 2, column = 0, padx = 80)
Entry_DOB = Entry(Frame_1, font = ('arial',15), width = 25, textvariable =
DOB)
Entry_DOB.grid(row = 2, column = 1, padx = 5, pady = 5)

Label_Gender = Label(Frame_1, text = 'Gender', font = ('arial',15,'bold'),
bg = 'Navajo white')
Label_Gender.grid(row = 2, column = 3, padx = 80)
Entry_Gender = ttk.Combobox(Frame_1, values = ('
','Male','Female','Others'), font = ('arial',15), width = 23, textvariable =
gender)
Entry_Gender.grid(row = 2, column = 4, padx = 5, pady = 5)

Label_School = Label(Frame_1, text = 'School Name', font =
('arial',15,'bold'), bg = 'Navajo white')
Label_School.grid(row = 3, column = 0, padx = 80)
Entry_School = Entry(Frame_1, font = ('arial',15), width = 25, textvariable
= scl)
Entry_School.grid(row = 3, column = 1, padx = 5, pady = 5)

Label_Email = Label(Frame_1, text = 'Email ID', font = ('arial',15,'bold'),
bg = 'Navajo white')
Label_Email.grid(row = 3, column = 3, padx = 80)
Entry_Email = Entry(Frame_1, font = ('arial',15), width = 25, textvariable =
email)
Entry_Email.grid(row = 3, column = 4, padx = 5, pady = 5)

```

```

#=====Frame_2=====
=====
Frame_2 = LabelFrame(root, width = 1200, height = 400, font =
('arial',20,'bold'), bg = 'Navajo white', bd = 10 \
, text = 'Grades Point Obtained', relief = 'ridge')
Frame_2.grid(row = 3, column = 0)

#=====Labels of
Frame_2=====

Label_Subject = Label(Frame_2, text = 'SUBJECT', font = ('arial',16,'bold'),
bg = 'Navajo white')
Label_Subject.grid(row = 3, column = 0, padx = 50, pady = 10)

Label_obt_Marks = Label(Frame_2, text = 'MARKS OBTAINED', font =
('arial',16,'bold'), bg = 'Navajo white')
Label_obt_Marks.grid(row = 3, column = 1, padx = 20)

Label_Subject = Label(Frame_2, text = 'PASSING MARKS', font =
('arial',16,'bold'), bg = 'Navajo white')
Label_Subject.grid(row = 3, column = 2, padx = 20)

Label_obt_Marks = Label(Frame_2, text = 'TOTAL MARKS', font =
('arial',16,'bold'), bg = 'Navajo white')
Label_obt_Marks.grid(row = 3, column = 3, padx = 20)

Label_1 = Label(Frame_2, text = 'MATHEMATICS', font = ('arial',14), bg =
'Navajo white')
Label_1.grid(row = 4, column = 0)
Label_2 = Label(Frame_2, text = 'PHYSICS', font = ('arial',14), bg = 'Navajo
white')
Label_2.grid(row = 5, column = 0)
Label_3 = Label(Frame_2, text = 'CHEMISTRY', font = ('arial',14), bg =
'Navajo white')
Label_3.grid(row = 6, column = 0)
Label_4 = Label(Frame_2, text = 'HINDI', font = ('arial',14), bg = 'Navajo
white')
Label_4.grid(row = 7, column = 0)
Label_5 = Label(Frame_2, text = 'ENGLISH', font = ('arial',14), bg = 'Navajo
white')
Label_5.grid(row = 8, column = 0)
Label_6 = Label(Frame_2, text = 'GRAND TOTAL', font = ('arial',16), bg =
'Navajo white')
Label_6.grid(row = 9, column = 0)
Label_7 = Label(Frame_2, text = 'PERCENTAGE', font = ('arial',16,'bold'), bg
= 'Navajo white')
Label_7.grid(row = 10, column = 0)
Label_8 = Label(Frame_2, text = 'CGPA', font = ('arial',16,'bold'), bg =
'Navajo white')
Label_8.grid(row = 10, column = 2)
Label_9 = Label(Frame_2, text = 'GRADE', font = ('arial',16,'bold'), bg =
'Navajo white')
Label_9.grid(row = 10, column = 4)
Label_10 = Label(Frame_2, text = 'DIVISION', font = ('arial',16,'bold'), bg
= 'Navajo white')
Label_10.grid(row = 11, column = 0)
Label_10 = Label(Frame_2, text = 'RESULT', font = ('arial',16,'bold'), bg =
'Navajo white')
Label_10.grid(row = 11, column = 2)

#=====Entries of
Frame_2=====
var_1 = StringVar(Frame_2, value = '33')
var_2 = StringVar(Frame_2, value = '100')
var_3 = StringVar(Frame_2, value = '500')

```

```

Entry__1 = Entry(Frame_2, font = ('arial',16), width = 5, textvariable = m1)
Entry__1.grid(row = 4, column = 1)
Entry__2 = Entry(Frame_2, font = ('arial',16), width = 5, textvariable = m2)
Entry__2.grid(row = 5, column = 1)
Entry__3 = Entry(Frame_2, font = ('arial',16), width = 5, textvariable = m3)
Entry__3.grid(row = 6, column = 1)
Entry__4 = Entry(Frame_2, font = ('arial',16), width = 5, textvariable = m4)
Entry__4.grid(row = 7, column = 1)
Entry__5 = Entry(Frame_2, font = ('arial',16), width = 5, textvariable = m5)
Entry__5.grid(row = 8, column = 1)
Entry__6 = Entry(Frame_2, font = ('arial',14), width = 5, textvariable = gt,
state = 'readonly')
Entry__6.grid(row = 9, column = 1, pady = 8)
Entry__7 = Entry(Frame_2, font = ('arial',14,'bold'), width = 5,
textvariable = per, state = 'readonly')
Entry__7.grid(row = 10, column = 1, pady = 8)
Entry__8 = Entry(Frame_2, font = ('arial',14,'bold'), width = 5,
textvariable = cgpa, state = 'readonly')
Entry__8.grid(row = 10, column = 3, pady = 8)
Entry__9 = Entry(Frame_2, font = ('arial',14,'bold'), width = 5,
textvariable = grade, state = 'readonly')
Entry__9.grid(row = 10, column = 5, padx = 20, pady = 8)
Entry__10 = Entry(Frame_2, font = ('arial',14,'bold'), width = 8,
textvariable = div, state = 'readonly')
Entry__10.grid(row = 11, column = 1, padx = 20, pady = 8)
Entry__11 = Entry(Frame_2, font = ('arial',14,'bold'), width = 7,
textvariable = result, state = 'readonly')
Entry__11.grid(row = 11, column = 3, padx = 20, pady = 8)

Entry_1_2 = Entry(Frame_2, textvariable = var_1, font = ('arial',16), width
= 5, state = 'readonly')
Entry_1_2.grid(row = 4, column = 2, pady = 5)
Entry_1_3 = Entry(Frame_2, textvariable = var_2, font = ('arial',16), width
= 5, state = 'readonly')
Entry_1_3.grid(row = 4, column = 3)
Entry_2_2 = Entry(Frame_2, textvariable = var_1, font = ('arial',16), width
= 5, state = 'readonly')
Entry_2_2.grid(row = 5, column = 2, pady = 5)
Entry_2_3 = Entry(Frame_2, textvariable = var_2, font = ('arial',16), width
= 5, state = 'readonly')
Entry_2_3.grid(row = 5, column = 3)
Entry_3_2 = Entry(Frame_2, textvariable = var_1, font = ('arial',16), width
= 5, state = 'readonly')
Entry_3_2.grid(row = 6, column = 2, pady = 5)
Entry_3_3 = Entry(Frame_2, textvariable = var_2, font = ('arial',16), width
= 5, state = 'readonly')
Entry_3_3.grid(row = 6, column = 3)
Entry_4_2 = Entry(Frame_2, textvariable = var_1, font = ('arial',16), width
= 5, state = 'readonly')
Entry_4_2.grid(row = 7, column = 2, pady = 5)
Entry_4_3 = Entry(Frame_2, textvariable = var_2, font = ('arial',16), width
= 5, state = 'readonly')
Entry_4_3.grid(row = 7, column = 3)
Entry_5_2 = Entry(Frame_2, textvariable = var_1, font = ('arial',16), width
= 5, state = 'readonly')
Entry_5_2.grid(row = 8, column = 2, pady = 5)
Entry_5_3 = Entry(Frame_2, textvariable = var_2, font = ('arial',16), width
= 5, state = 'readonly')
Entry_5_3.grid(row = 8, column = 3)
Entry_6_3 = Entry(Frame_2, textvariable = var_3, font = ('arial',16), width
= 5, state = 'readonly')
Entry_6_3.grid(row = 9, column = 3)

```

```

#=====Buttons=====

```

```

Btn_Compute = Button(Frame_2, text = 'COMPUTE', font = ('arial',12,'bold'),

```

```

width = 10, command = Compute)
    Btn_Compute.grid(row = 4, column = 4, padx = 50, pady = 6)
    Btn_Save = Button(Frame_2, text = 'SAVE', font = ('arial',12,'bold'), width
= 10, command = Add)
    Btn_Save.grid(row = 5, column = 4, padx = 50, pady = 6)
    Btn_Update = Button(Frame_2, text = 'UPDATE', font = ('arial',12,'bold'),
width = 10, command = Update)
    Btn_Update.grid(row = 6, column = 4, padx = 50, pady = 6)
    Btn_Cancel = Button(Frame_2, text = 'RESET', font = ('arial',12,'bold'),
width = 10, command = Reset)
    Btn_Cancel.grid(row = 7, column = 4, padx = 50, pady = 6)
    Btn_Exit = Button(Frame_2, text = 'EXIT', font = ('arial',12,'bold'), width
= 10, command = Exit)
    Btn_Exit.grid(row = 8, column = 4, padx = 50, pady = 6)

```

```

root.mainloop()

```

```

def search_result_marksheet(row):
    root = Tk()
    root.title('Marksheet')
    root.geometry('1350x750')
    root.config(bg = 'Navajo white')

```

```

#=====Functions=====
=====

```

```

def Compute():
    x1 = (m1.get());      x2 = (m2.get());      x3 = (m3.get());      x4 =
(m4.get());      x5 = (m5.get())

    tot = x1+x2+x3+x4+x5
    gt.set(tot)

    Per = ((x1+x2+x3+x4+x5) * 100)/500
    per.set(Per)

    cg = (((x1+x2+x3+x4+x5) * 100)/500) / 9.5
    cgra.set(round(cg,1))

    if (((x1+x2+x3+x4+x5) * 100)/500) <= 40:
        grd = 'G'
    elif (((x1+x2+x3+x4+x5) * 100)/500) <= 50:
        grd = 'F'
    elif (((x1+x2+x3+x4+x5) * 100)/500) <= 60:
        grd = 'E'
    elif (((x1+x2+x3+x4+x5) * 100)/500) <= 70:
        grd = 'D'
    elif (((x1+x2+x3+x4+x5) * 100)/500) <= 80:
        grd = 'C'
    elif (((x1+x2+x3+x4+x5) * 100)/500) <= 90:
        grd = 'B'
    else:
        grd = 'A'

    grade.set(grd)

    count = 0
    if x1 < 33:
        count = count + 1
    if x2 < 33:
        count = count + 1
    if x3 < 33:
        count = count + 1

```

```

        if x4 < 33:
            count = count + 1
        if x5 < 33:
            count = count + 1

        if (count == 0):
            result.set('PASS')
        elif (count == 1 or count == 2 ):
            result.set('SUPPLY')
        else:
            result.set('FAIL')

        if Per <= 45 and result != "FAIL":
            div.set('THIRD')
        elif Per <= 60 and result != "FAIL":
            div.set('SECOND')
        elif Per <= 100:
            div.set('FIRST')

#=====Frame_1=====
=====

        Frame_1 = LabelFrame(root, width = 1200, height = 400, font =
('arial',20,'bold'), bg = 'Navajo white', bd = 10, \
            text = 'Student Details', relief = 'ridge')
        Frame_1.grid(row = 1, column = 0, pady = 20, padx = 20)

        name = StringVar(Frame_1,value=row[0][1])
        roll = StringVar(Frame_1,value=row[0][2])
        fname = StringVar(Frame_1,value=row[0][3])
        mname = StringVar(Frame_1,value=row[0][4])
        DOB = StringVar(Frame_1,value=row[0][5])
        gender = StringVar(Frame_1,value=row[0][6])
        scl = StringVar(Frame_1,value=row[0][7])
        email = StringVar(Frame_1,value=row[0][8])

#=====Labels and Entries for
Frame_1=====
        Label_Name = Label(Frame_1, text = 'Name', font = ('arial',15,'bold'), bg =
'Navajo white')
        Label_Name.grid(row = 0, column = 0, padx = 80)
        Entry_Name = Entry(Frame_1, font = ('arial',15), width = 25, textvariable =
name)
        Entry_Name.grid(row = 0, column = 1, padx = 5, pady = 5)

        Label_Roll_no = Label(Frame_1, text = 'Roll Number', font =
('arial',15,'bold'), bg = 'Navajo white')
        Label_Roll_no.grid(row = 0, column = 3, padx = 80)
        Entry_Roll_no = Entry(Frame_1, font = ('arial',15), width = 25, textvariable
= roll)
        Entry_Roll_no.grid(row = 0, column = 4, padx = 40)

        Label_Father_Name = Label(Frame_1, text = 'Father Name', font =
('arial',15,'bold'), bg = 'Navajo white')
        Label_Father_Name.grid(row = 1, column = 0, padx = 80)
        Entry_Father_Name = Entry(Frame_1, font = ('arial',15), width = 25,
textvariable = fname)
        Entry_Father_Name.grid(row = 1, column = 1, padx = 5, pady = 10)

        Label_Mother_Name = Label(Frame_1, text = 'Mother Name', font =
('arial',15,'bold'), bg = 'Navajo white')
        Label_Mother_Name.grid(row = 1, column = 3, padx = 80)
        Entry_Mother_Name = Entry(Frame_1, font = ('arial',15), width = 25,

```

```

textvariable = mname)
    Entry_Mother_Name.grid(row = 1, column = 4, padx = 5)

    Label_DOB = Label(Frame_1, text = 'Date of Birth', font =
('arial',15,'bold'), bg = 'Navajo white')
    Label_DOB.grid(row = 2, column = 0, padx = 80)
    Entry_DOB = Entry(Frame_1, font = ('arial',15), width = 25, textvariable =
DOB)
    Entry_DOB.grid(row = 2, column = 1, padx = 5, pady = 5)

    Label_Gender = Label(Frame_1, text = 'Gender', font = ('arial',15,'bold'),
bg = 'Navajo white')
    Label_Gender.grid(row = 2, column = 3, padx = 80)
    Entry_Gender = Entry(Frame_1, font = ('arial',15), width = 25, textvariable
= gender)
    Entry_Gender.grid(row = 2, column = 4, padx = 5, pady = 5)

    Label_School = Label(Frame_1, text = 'School Name', font =
('arial',15,'bold'), bg = 'Navajo white')
    Label_School.grid(row = 3, column = 0, padx = 80)
    Entry_School = Entry(Frame_1, font = ('arial',15), width = 25, textvariable
= scl)
    Entry_School.grid(row = 3, column = 1, padx = 5, pady = 5)

    Label_Email = Label(Frame_1, text = 'Email ID', font = ('arial',15,'bold'),
bg = 'Navajo white')
    Label_Email.grid(row = 3, column = 3, padx = 80)
    Entry_Email = Entry(Frame_1, font = ('arial',15), width = 25, textvariable =
email)
    Entry_Email.grid(row = 3, column = 4, padx = 5, pady = 5)

#=====Frame_2=====
#=====
Frame_2 = LabelFrame(root, width = 1200, height = 400, font =
('arial',20,'bold'), bg = 'Navajo white', bd = 10 \
, text = 'Grades Point Obtained', relief = 'ridge')
Frame_2.grid(row = 3, column = 0)

m1 = DoubleVar(Frame_2,row[0][9])
m2 = DoubleVar(Frame_2,row[0][10])
m3 = DoubleVar(Frame_2,row[0][11])
m4 = DoubleVar(Frame_2,row[0][12])
m5 = DoubleVar(Frame_2,row[0][13])
gt = DoubleVar(Frame_2,row[0][14])
per = DoubleVar(Frame_2,row[0][15])
cgpa = DoubleVar(Frame_2,row[0][16])
grade = StringVar(Frame_2,row[0][17])
div = StringVar(Frame_2,row[0][18])
result = StringVar(Frame_2,row[0][19])

#=====Labels of
Frame_2=====
Label_Subject = Label(Frame_2, text = 'SUBJECT', font = ('arial',16,'bold'),
bg = 'Navajo white')
Label_Subject.grid(row = 3, column = 0, padx = 50, pady = 10)

Label_obt_Marks = Label(Frame_2, text = 'MARKS OBTAINED', font =
('arial',16,'bold'), bg = 'Navajo white')
Label_obt_Marks.grid(row = 3, column = 1, padx = 20)

Label_Subject = Label(Frame_2, text = 'PASSING MARKS', font =
('arial',16,'bold'), bg = 'Navajo white')
Label_Subject.grid(row = 3, column = 2, padx = 20)

```

```

Label_obt_Marks = Label(Frame_2, text = 'TOTAL MARKS', font =
('arial',16,'bold'), bg = 'Navajo white')
Label_obt_Marks.grid(row = 3, column = 3, padx = 20)

Label_1 = Label(Frame_2, text = 'MATHEMATICS', font = ('arial',14), bg =
'Navajo white')
Label_1.grid(row = 4, column = 0)
Label_2 = Label(Frame_2, text = 'PHYSICS', font = ('arial',14), bg = 'Navajo
white')
Label_2.grid(row = 5, column = 0)
Label_3 = Label(Frame_2, text = 'CHEMISTRY', font = ('arial',14), bg =
'Navajo white')
Label_3.grid(row = 6, column = 0)
Label_4 = Label(Frame_2, text = 'HINDI', font = ('arial',14), bg = 'Navajo
white')
Label_4.grid(row = 7, column = 0)
Label_5 = Label(Frame_2, text = 'ENGLISH', font = ('arial',14), bg = 'Navajo
white')
Label_5.grid(row = 8, column = 0)
Label_6 = Label(Frame_2, text = 'GRAND TOTAL', font = ('arial',16), bg =
'Navajo white')
Label_6.grid(row = 9, column = 0)
Label_7 = Label(Frame_2, text = 'PERCENTAGE', font = ('arial',16,'bold'), bg
= 'Navajo white')
Label_7.grid(row = 10, column = 0)
Label_8 = Label(Frame_2, text = 'CGPA', font = ('arial',16,'bold'), bg =
'Navajo white')
Label_8.grid(row = 10, column = 2)
Label_9 = Label(Frame_2, text = 'GRADE', font = ('arial',16,'bold'), bg =
'Navajo white')
Label_9.grid(row = 10, column = 4)
Label_10 = Label(Frame_2, text = 'DIVISION', font = ('arial',16,'bold'), bg
= 'Navajo white')
Label_10.grid(row = 11, column = 0)
Label_10 = Label(Frame_2, text = 'RESULT', font = ('arial',16,'bold'), bg =
'Navajo white')
Label_10.grid(row = 11, column = 2)

#=====Entries of
Frame_2=====
var_1 = StringVar(Frame_2, value = '33')
var_2 = StringVar(Frame_2, value = '100')
var_3 = StringVar(Frame_2, value = '500')

Entry__1 = Entry(Frame_2, font = ('arial',16), width = 5, textvariable = m1)
Entry__1.grid(row = 4, column = 1)
Entry__2 = Entry(Frame_2, font = ('arial',16), width = 5, textvariable = m2)
Entry__2.grid(row = 5, column = 1)
Entry__3 = Entry(Frame_2, font = ('arial',16), width = 5, textvariable = m3)
Entry__3.grid(row = 6, column = 1)
Entry__4 = Entry(Frame_2, font = ('arial',16), width = 5, textvariable = m4)
Entry__4.grid(row = 7, column = 1)
Entry__5 = Entry(Frame_2, font = ('arial',16), width = 5, textvariable = m5)
Entry__5.grid(row = 8, column = 1)
Entry__6 = Entry(Frame_2, font = ('arial',14), width = 5, textvariable = gt)
Entry__6.grid(row = 9, column = 1, pady = 8)
Entry__7 = Entry(Frame_2, font = ('arial',14,'bold'), width = 5,
textvariable = per)
Entry__7.grid(row = 10, column = 1, pady = 8)
Entry__8 = Entry(Frame_2, font = ('arial',14,'bold'), width = 5,
textvariable = cgpa)
Entry__8.grid(row = 10, column = 3, pady = 8)
Entry__9 = Entry(Frame_2, font = ('arial',14,'bold'), width = 5,
textvariable = grade)
Entry__9.grid(row = 10, column = 5, padx = 20, pady = 8)
Entry__10 = Entry(Frame_2, font = ('arial',14,'bold'), width = 8,
textvariable = div)
Entry__10.grid(row = 11, column = 1, padx = 20, pady = 8)
Entry__11 = Entry(Frame_2, font = ('arial',14,'bold'), width = 7,

```

```

textvariable = result)
    Entry_11.grid(row = 11, column = 3, padx = 20, pady = 8)

    Entry_1_2 = Entry(Frame_2, textvariable = var_1, font = ('arial',16), width
= 5)
    Entry_1_2.grid(row = 4, column = 2, pady = 5)
    Entry_1_3 = Entry(Frame_2, textvariable = var_2, font = ('arial',16), width
= 5)
    Entry_1_3.grid(row = 4, column = 3)
    Entry_2_2 = Entry(Frame_2, textvariable = var_1, font = ('arial',16), width
= 5)
    Entry_2_2.grid(row = 5, column = 2, pady = 5)
    Entry_2_3 = Entry(Frame_2, textvariable = var_2, font = ('arial',16), width
= 5)
    Entry_2_3.grid(row = 5, column = 3)
    Entry_3_2 = Entry(Frame_2, textvariable = var_1, font = ('arial',16), width
= 5)
    Entry_3_2.grid(row = 6, column = 2, pady = 5)
    Entry_3_3 = Entry(Frame_2, textvariable = var_2, font = ('arial',16), width
= 5)
    Entry_3_3.grid(row = 6, column = 3)
    Entry_4_2 = Entry(Frame_2, textvariable = var_1, font = ('arial',16), width
= 5)
    Entry_4_2.grid(row = 7, column = 2, pady = 5)
    Entry_4_3 = Entry(Frame_2, textvariable = var_2, font = ('arial',16), width
= 5)
    Entry_4_3.grid(row = 7, column = 3)
    Entry_5_2 = Entry(Frame_2, textvariable = var_1, font = ('arial',16), width
= 5)
    Entry_5_2.grid(row = 8, column = 2, pady = 5)
    Entry_5_3 = Entry(Frame_2, textvariable = var_2, font = ('arial',16), width
= 5)
    Entry_5_3.grid(row = 8, column = 3)
    Entry_6_3 = Entry(Frame_2, textvariable = var_3, font = ('arial',16), width
= 5)
    Entry_6_3.grid(row = 9, column = 3)

#=====Buttons=====
=====

    Btn_Exit = Button(Frame_2, text = 'EXIT', font = ('arial',12,'bold'), width
= 10, command = root.destroy)
    Btn_Exit.grid(row = 8, column = 4, padx = 50, pady = 6)

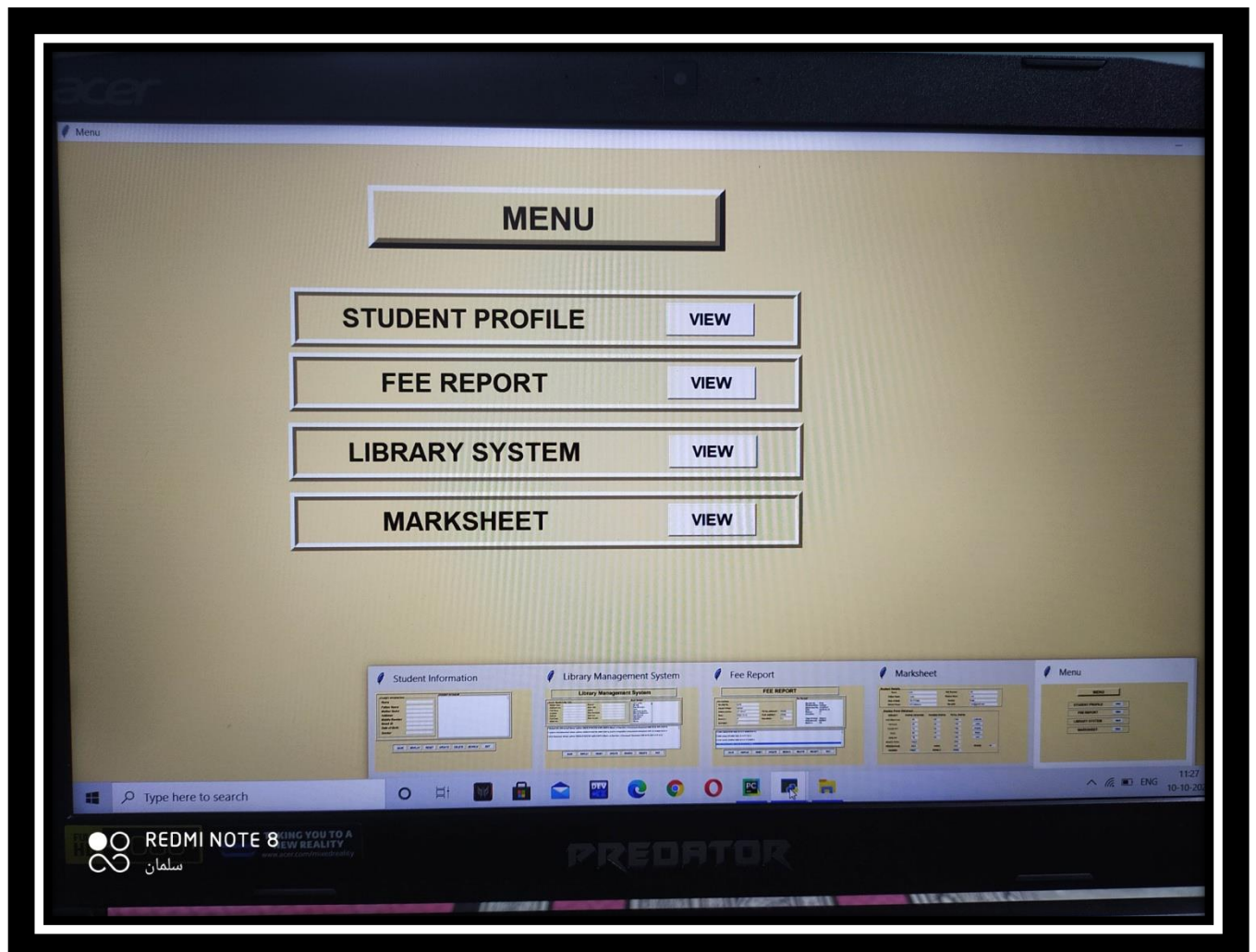
    root.mainloop()

if __name__ == '__main__':
    marksheet()

```

b. Screenshots:-

1. Menu



2. Student Profile

Student Information

STUDENT INFORMATION

Name

Father Name

Mother Name

Address

Mobile Number

Email ID

Date of Birth

Gender

STUDENT DATABASE

STUDENT ID	NAME	FATHER NAME	MOTHER NAME	ADDRESS	MOBILE NUMBER	EMAIL ID	DATE OF BIRTH	GENDER
------------	------	-------------	-------------	---------	---------------	----------	---------------	--------

SAVE

DISPLAY

RESET

UPDATE

DELETE

SEARCH

EXIT

REDMI NOTE 8
Type here to search
سماں

TAKING YOU TO A NEW REALITY
www.mi.com/redminote8

PREDATOR

3. Fee Report

Fee Report

FEE REPORT

Informations

Receipt No. : 2345

Student Name : salman

Admission No. : 37120047

Date : 2020-10-10

Branch : IT

Semester : 7

TOTAL AMOUNT : 36265

PAID AMOUNT : 36260

BALANCE : 5.0

Fee Receipt

Receipt No. : 2345

Student Name : salman

Admission No. : 37120047

Date : 2020-10-10

Branch : IT

Semester : 7

Total Amount : 36265.0

Paid Amount : 36260.0

Balance : 5.0

1 12345 salman 87956 2020-10-10 IT SEVENTH 0 0 0

2 2345 salman 37120047 2020-10-10 IT 7 0 0 0

3 2345 salman 37120047 2020-10-10 IT 7 0 36260 0

4 2345 salman 37120047 2020-10-10 IT 7 0 36260 0

SAVE

DISPLAY

RESET

UPDATE

SEARCH

DELETE

RECEIPT

EXIT

REDMI NOTE 8 سلمان

PREDATOR

11:15 10-10-2020

4. Library System

Library Management System

Library Management System

Library Membership Info:

Member type		Book ID	
Reference No.		Book Title	
First Name		Author	
Surname		Date Borrowed	
Address		Date Due	
Post Code		Days in Loan	
Mobile No.			

Book Details:

Java Script	
My SQL	
Data Structure	
Linux	
Operating System	
Web Development	
Data Science	
Algorithms	
Android	
VB.net	

1	Student	2541	Mohammed	Salman	oddisha	768036	9178932786	{ISBN 535674}	{Basics of Algorithm }	{Karumanchi Narasimha}	2020-10-10	2020-10-20	10
2	Student	2541	Mohammed	Salman	oddisha	768036	9178932786	{ISBN 535674}	{Basics of Algorithm }	{Karumanchi Narasimha}	2020-10-10	2020-10-20	10
3	2541	Mohammed	Salman	oddisha	768036	9178932786	{ISBN 535674}	{Basics of Algorithm }	{Karumanchi Narasimha}	2020-10-10	2020-10-20	10	}

SAVE

DISPLAY

RESET

UPDATE

SEARCH

DELETE

EXIT

REDMI NOTE 8
سلمان
BRING YOU TO A NEW REALITY
PREDATOR

5 Marksheet

Marksheet

Student Details

Name	ram	Roll Number	92
Father Name	ramu	Mother Name	rani
Date of Birth	20/12/1999	Gender	Male
School Name	sri chaitaniya	Email ID	ram@gmail.com

Grades Point Obtained

SUBJECT	MARKS OBTAINED	PASSING MARKS	TOTAL MARKS	
MATHEMATICS	89	33	100	COMPUTE
PHYSICS	95	33	100	SAVE
CHEMISTRY	79	33	100	UPDATE
HINDI	88	33	100	RESET
ENGLISH	78	33	100	EXIT
GRAND TOTAL	429.0		500	
PERCENTAGE	85.8	CGPA	9.0	GRADE B
DIVISION	FIRST	RESULT	PASS	

REDMI NOTE 8
سلمان
TAKING YOU TO A NEW REALITY
PREDATOR