**E-BOOK MANAGEMENT SYSTEM**

**A MINI PROJECT REPORT**

***Submitted by***

**KARTHICK S P**

**NANDHINI VENKATESAN**

**GOPIKA P**

**SOWMIYA A S**

***in partial fulfilment for the award of the***

***degree of***

**BACHELOR OF TECHNOLOGY**

***In***

**INFORMATION TECHNOLOGY**

**KNOWLEDGE INSTITUTE OF TECHNOLOGY,**

**SALEM- 637 504**

**MAY-2023**

**BONAFIDE CERTIFICATE**

Certified that this project report **“E-BOOK MANAGEMENT SYSTEM”** is the bonafide work of “**KARTHICK S P (611220205015), NANDHINI VENKATESAN (611220205020), GOPIKA P (611220205303), SOWMIYA A S (611220205036)”** who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

**SIGNATURE SIGNATURE**

**Mr. M. SIVARAMAN Dr.P.SACHIDHANANDHAM,ME.,Ph.d.,**

**ASSISTANT PROFESSOR, HEAD OF THE DEPARTMENT**

**Deprtment of INFORMATION Department of INFORMATION**

**TECHNOLOGY, TECHNOLOGY, Knowledge Institute of Knowledge Institute of**

**Technology , Technology,**

**Kakapalayam,Salem-637 504. Kakapalyam ,Salem-637 504.**

**Submitted for Semester Mini-Project viva-voce examination held on**

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**ACKNOWLEDGEMENT**

At the outset, we express our heartfelt gratitude to God, who has been our strength to bring this mini project to light.

At this pleasing moment of having successfully completed our project, we wish to convey our sincere thanks and gratitude to our beloved president

**Mr. C. BALAKRISHNAN,** who has provided all the facilities to us

We would like to convey our sincere thanks to our beloved Principal, **Dr.PSS. SRINIVASAN**, who forward us to do our mini project and offers adequate duration to complete our mini project.

We express our sincere thanks to **Dr. P. SACHIDHANANDHAM**, Head of the Department, Department of Computer Information Technology, for fostering the excellent academic climate in the department.

We express our pronounced sense of thanks with deepest respect and gratitude to our supervisor **Mrs. G. USHA,** Assistant Professor Department of Information Technology, for her valuable and precious guidance and for having amicable relation.

With deep sense of gratitude, we extend our earnest and sincere thanks to our mini project coordinator supervisor **Mr. M. SIVARAMAN**, Assistant Professor Department of Information Technology, for her guidance and encouragement during this mini project.

We would also like to express our thanks to all the staff members of our department, friends and students who helped us directly and indirectly in all aspects of the mini project work to get completed successfully.

###### **TABLE OF CONTENTS**

**CHAPTER NO. TITLE PAGE NO. ABSTARCT 5**

**LIST OF FIGURES 6**

1. **INTRODUCTION** 7
2. **SYSTEM ANALYSIS 9**
   1. EXISTING SYSTEM
   2. PROBLEM STATEMENT
   3. PROPOSED SYSTEM
3. **SYSTEM DESIGN 12**
4. **SYSTEM IMPLEMENTATION 16**
5. **CONCLUSION 36**

**ABSTRACT**

**E-BOOK MANAGEMENT SYSTEM** is a project which aims in developing a computerized system to maintain all the daily work of E-Book .A e-Book management system is a system that is used to maintain the records of the e-book. The record includes reader and author who borrowed books from our e-book management application.

The e-book management system only can be used by the admin who can add new member like reader or author in the system along with their details. The admin can also update and delete members in the system. The e-book management system stores the detail of the member in the database.

The system also provides the book detail by just clicking on the name or title of the book which is already present in the book box.

When a member request a book to buy, the detail of the book is add to the members detail.

**CHAPTER NO 1**

**INTRODUCTION**

The “**E-BOOK MANAGEMENT SYSTEM**” is developed with the help of “**PYTHON PROGRAMMING LANGUAGE**”. The frontend of the software or system is developed with the help of “**PYTHON TKINTER”** and the backend of the software is developed with the help of “**MYSQL WORKBENCH 8.0**”.

**PYTHON TKINTER:**

Tkinter is the defacto way in Python to **create Graphical User interfaces (GUIs)** and is included in all standard Python Distributions. In fact, it's the only framework built into the Python standard library.

Out of all the GUI methods, tkinter is the most commonly used method. **It is a standard Python interface to the Tk GUI toolkit shipped with Python**. Python with tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.

There are a number of widgets which you can put in your tkinter application. Some of the widgets that are used in this application are explained below:

1. **Button**: To add a button in your application, this widget is used.
2. **Entry:** It is used to input the single line text entry from the user. For multi-line text input, Text widget is used.
3. **Frame:** It acts as a container to hold the widgets. It is used for grouping and organizing the widgets.
4. **Label**: It refers to the display box where you can put any text or image which can be updated any time as per the code.
5. **Listbox**: It offers a list to the user from which the user can accept any number of options.
6. **Scrollbar**: It refers to the slide controller which will be used to implement listed widgets.
7. **Message**: It refers to the multi-line and non-editable text. It works same as that of Label.

**MYSQL WORKBENCH:**

MySQL Workbench is a visual database design tool that integrates SQL development, administration, database design, creation and maintenance into a single integrated development environment for the MySQL database system.

The library management system can “Add”, “Delete” and “update” members in the system. The library management system use SQL queries to add, delete and update member.There are three types of members “Teacher”, “Student” and “Admin”. The library management system alsoshows the detail of the member and the book by just clicking on them.

The library management system was developed on “**visual studio code**”and the VS code was connected with MySQL workbench with the help of “**MySQL connector”**function

**CHAPTER NO 2**

**SYSTEM ANALYSIS**

* 1. **EXISTING SYSTEM**
* In the current existing system, one must manually keep the record of each and all books and members.
* Which take too much time and it is a very long process.
  1. **PROBLEM STATEMEN**
* The problem in the system is that you have to enter the data of the book and member manually and it too much time.
* The data or the detail of the book may include book id, name, price ,due date ,borrowed date, fine amount and actual price of the book.
* While borrowing the book the librarian has to enter the due date manually which may be confusing.
* The librarian can’t show the full detail of member.
  1. **PROPOSED SYSTEM**
* In our system once the data of the member is added for the next the you have to just update the data.
* If a member borrows a new book the librarian just needs to click on the members data along with the book name the member borrowed and just click update button to update the data.
* The librarian can show full data of the member along with book he or she borrowed.
* The system automatically calculate the due date and time form the current date and time.
* The librarian can also delete the member if he or she is no longer part of the library.

**CHAPTER 3**

**SYSTEM DESIGN**

UML DIAGRAMS

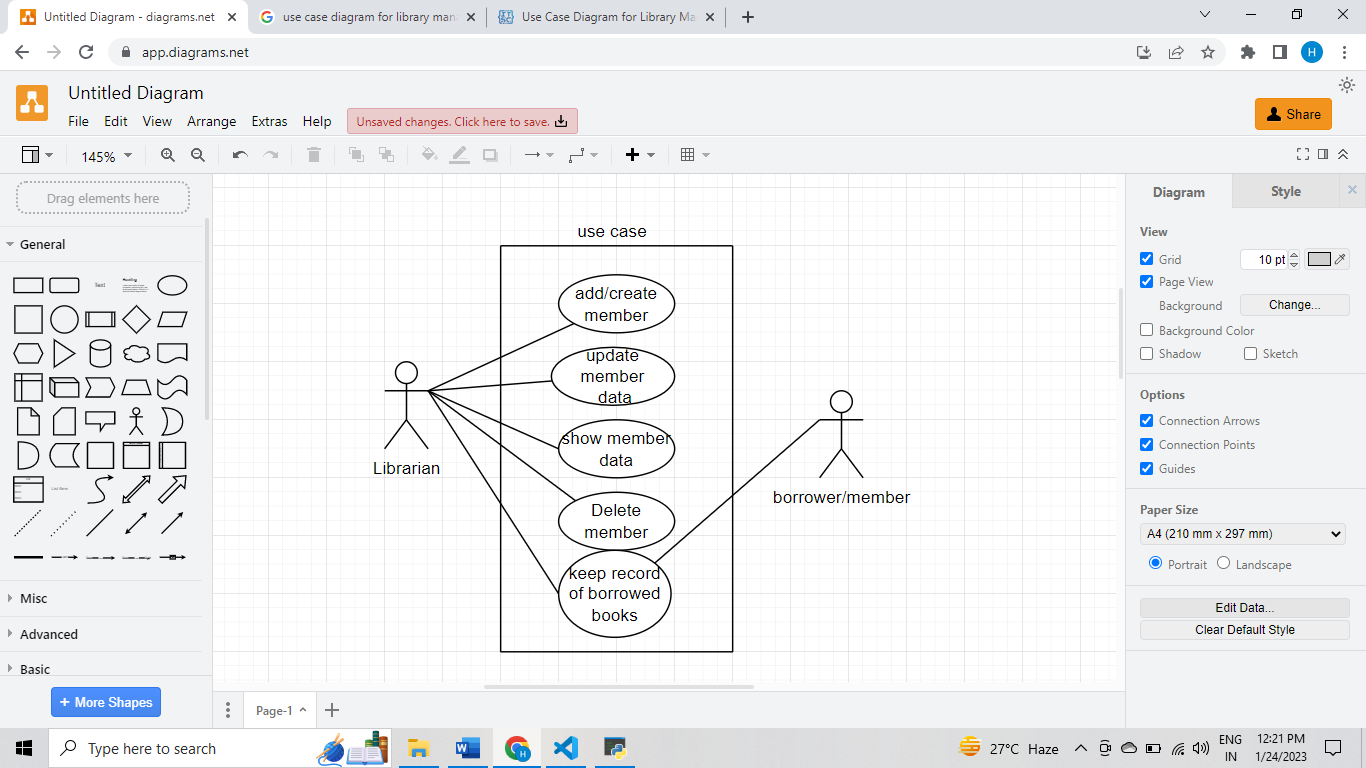
LIST OF FIGURES

FIGURE NO. NAME OF FIGURE PAGE NO.

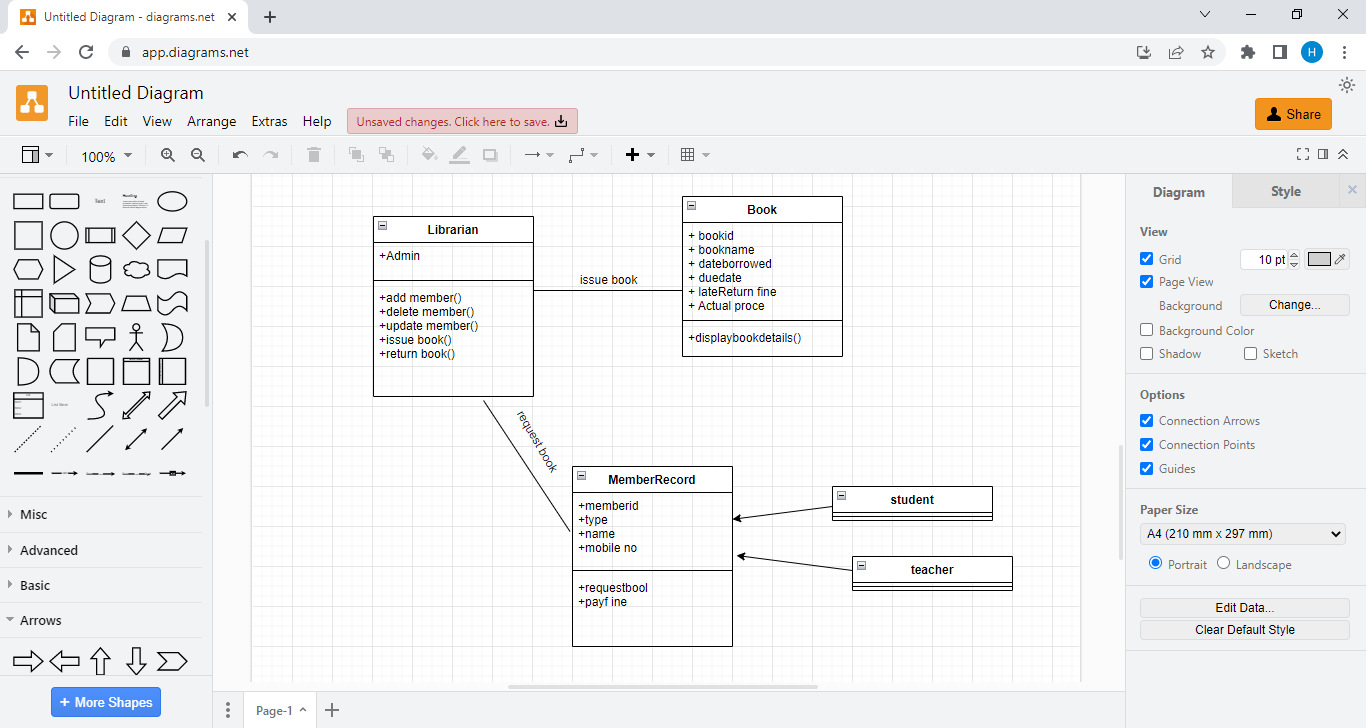
3.1.1 USECASE DIAGRAM 12

3.1.2 CLASS DIAGRAM 13

3.1USECASE DIAGRAM



3.2 CLASS DIAGRAM



**CHAPTER 4**

**SYSTEM IMPLEMENTATION**

from tkinter import\*

from tkinter import ttk

import tkinter as tk

import mysql.connector

from tkinter import messagebox

import datetime

import tkinter

class LibraryManagementSystem:

def \_\_init\_\_(self,root):

self.root=root

self.root.title("LibraryManagementSystem")

self.root.geometry("1200x900+0+0")

#-----------------------------------Variable----------------------------------------------------------------

self.Member\_var=StringVar()

self.ID\_number\_var=StringVar()

self.Department\_var=StringVar()

self.Year\_var=StringVar()

self.Name\_var=StringVar()

self.Mobile\_no\_var=StringVar()

self.Book\_ID\_var=StringVar()

self.Book\_Name\_var=StringVar()

self.Date\_Borrowed\_var=StringVar()

self.Due\_Date\_var=StringVar()

self.Late\_Return\_var=StringVar()

self.Date\_Over\_Due\_var=StringVar()

self.ActualPrice\_var=StringVar()

def idata():

conn=mysql.connector.connect(host="localhost",username="root",password="mysql",database="librarymanagementsystem")

my\_cursor=conn.cursor()

my\_cursor.execute("insert into library values(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)",

(

self.Member\_var.get(),

self.ID\_number\_var.get(),

self.Department\_var.get(),

self.Year\_var.get(),

self.Name\_var.get(),

self.Mobile\_no\_var.get(),

self.Book\_ID\_var.get(),

self.Book\_Name\_var.get(),

self.Date\_Borrowed\_var.get(),

self.Due\_Date\_var.get(),

self.Late\_Return\_var.get(),

self.Date\_Over\_Due\_var.get(),

self.ActualPrice\_var.get()

))

conn.commit()

self.fatch\_data()

conn.close()

messagebox.showinfo("Success","Member has be inserted sucessfully")

#------------------------------------------------ creating a lable ------------------------------------------------------------------#

lbltitle=Label(self.root,text="Library Management System",bg="powder blue",fg="black",bd=15,relief=RIDGE,font=("times new roman",50,"bold"),padx=2,pady=6)

lbltitle.pack(side=TOP,fill=X)

#-------------------------------------------------- creating a frame -----------------------------------------------------------------#

frame=Frame(self.root,bd=12,relief=RIDGE,padx=15,bg="powder blue")

frame.place(x=0,y=120,width=1370,height=400)

#------------------------------------------- creating frames inside a frame ----------------------------------------------------------#

#------------------------------------------- DataFrameLeft ---------------------------------------------------------------------------#

DataFrameLeft=LabelFrame(frame,text="Library member",bg="powder blue",fg="black",bd=10,relief=RIDGE,font=("times new roman",15,"bold"))

DataFrameLeft.place(x=0,y=5,width=800,height=350)

lblMember=Label(DataFrameLeft,bg="powder blue",text="Member Type",font=("times new roman",15,"bold"),padx=2,pady=6)

lblMember.grid(row=0,column=0,sticky=W)

comMember=ttk.Combobox(font=("times new roman",15,"bold"),textvariable=self.Member\_var,width=20,state="readonly")

comMember["value"]=("Admin","Student","Teacher")

comMember.pack(padx=180,pady=50,anchor=SW)

lblID\_number=Label(DataFrameLeft,bg="powder blue" ,text="ID\_number",font=("times new roman",13,"bold"))

lblID\_number.grid(row=1,column=0,sticky=W)

txtID=Entry(DataFrameLeft,font=("times new roman",15,"bold"),textvariable=self.ID\_number\_var,width=20)

txtID.grid(row=1,column=1,padx=15)

lblDepartment=Label(DataFrameLeft,bg="powder blue" ,text="Department",font=("times new roman",13,"bold"))

lblDepartment.grid(row=2,column=0,sticky=W)

txtDepartment=Entry(DataFrameLeft,font=("times new roman",15,"bold"),textvariable=self.Department\_var,width=20)

txtDepartment.grid(row=2,column=1,padx=16,pady=5)

lblYEAR=Label(DataFrameLeft,bg="powder blue" ,text="YEAR",font=("times new roman",13,"bold"))

lblYEAR.grid(row=3,column=0,sticky=W)

comYEAR=ttk.Combobox(font=("times new roman",15,"bold"),textvariable=self.Year\_var,width=20,state="readonly")

comYEAR["value"]=("Staff","I","II","III","IV")

comYEAR.pack(padx=180,pady=20,anchor=SW)

lblName=Label(DataFrameLeft,bg="powder blue" ,text="Name",font=("times new roman",13,"bold"))

lblName.grid(row=4,column=0,sticky=W)

txtName=Entry(DataFrameLeft,font=("times new roman",15,"bold"),textvariable=self.Name\_var,width=20)

txtName.grid(row=4,column=1,padx=16,pady=9)

lblMobile\_no=Label(DataFrameLeft,bg="powder blue" ,text="Mobile\_no",font=("times new roman",13,"bold"))

lblMobile\_no.grid(row=5,column=0,sticky=W)

txtMobile\_no=Entry(DataFrameLeft,font=("times new roman",15,"bold"),textvariable=self.Mobile\_no\_var,width=20)

txtMobile\_no.grid(row=5,column=1,padx=20,pady=2)

lblBook\_Id=Label(DataFrameLeft,bg="powder blue" ,text="Book\_Id",font=("times new roman",13,"bold"))

lblBook\_Id.grid(row=6,column=0,sticky=W)

txtBook\_Id=Entry(DataFrameLeft,font=("times new roman",15,"bold"),textvariable=self.Book\_ID\_var,width=20)

txtBook\_Id.grid(row=6,column=1,padx=20,pady=2)

lblBook\_Name=Label(DataFrameLeft,bg="powder blue" ,text="Book\_Name",font=("times new roman",13,"bold"))

lblBook\_Name.grid(row=7,column=0,sticky=W)

txtBook\_Name=Entry(DataFrameLeft,font=("times new roman",15,"bold"),textvariable=self.Book\_Name\_var,width=20)

txtBook\_Name.grid(row=7,column=1,padx=20,pady=2)

lblDateBorrowed=Label(DataFrameLeft,bg="powder blue" ,text="Date\_Borrowed",font=("times new roman",13,"bold"))

lblDateBorrowed.grid(row=8,column=0,sticky=W)

txtDateBorrowed=Entry(DataFrameLeft,font=("times new roman",15,"bold"),textvariable=self.Date\_Borrowed\_var,width=20)

txtDateBorrowed.grid(row=8,column=1,padx=20,pady=2)

lblDateDue=Label(DataFrameLeft,bg="powder blue" ,text="Date\_Due",font=("times new roman",13,"bold"))

lblDateDue.grid(row=0,column=2,sticky=W)

txtDateDue=Entry(DataFrameLeft,font=("times new roman",15,"bold"),textvariable=self.Due\_Date\_var,width=20)

txtDateDue.grid(row=0,column=2,padx=190,pady=2)

lblLate\_Return\_fine=Label(DataFrameLeft,bg="powder blue" ,text="Late\_Return\_fine",font=("times new roman",13,"bold"))

lblLate\_Return\_fine.grid(row=1,column=2,sticky=W)

txtLate\_Return\_fine=Entry(DataFrameLeft,font=("times new roman",15,"bold"),textvariable=self.Late\_Return\_var,width=20)

txtLate\_Return\_fine.grid(row=1,column=2,padx=190,pady=2)

lblDateOverDue=Label(DataFrameLeft,bg="powder blue" ,text="Date\_Over\_Due",font=("times new roman",13,"bold"))

lblDateOverDue.grid(row=2,column=2,sticky=W)

txtDateOverDue=Entry(DataFrameLeft,font=("times new roman",15,"bold"),textvariable=self.Date\_Over\_Due\_var,width=20)

txtDateOverDue.grid(row=2,column=2,padx=90,pady=2)

lblActualPrice=Label(DataFrameLeft,bg="powder blue" ,text="ActualPrice",font=("times new roman",13,"bold"))

lblActualPrice.grid(row=3,column=2,sticky=W)

txtActualPrice=Entry(DataFrameLeft,font=("times new roman",15,"bold"),textvariable=self.ActualPrice\_var,width=20)

txtActualPrice.grid(row=3,column=2,padx=90,pady=2)

#------------------------------------------- DataFrameRight --------------------------------------------------------------------------#

DataFrameRight=LabelFrame(frame,text="BOOKS",bg="powder blue",fg="black",bd=10,relief=RIDGE,font=("times new roman",15,"bold"))

DataFrameRight.place(x=810,y=5,width=500,height=350)

self.textbox=Text(DataFrameRight,font=("times new roman",15,"bold"),width=23,height=13,padx=5,pady=2)

self.textbox.grid(row=0,column=2)

listScrollbar=Scrollbar(DataFrameRight)

listScrollbar.grid(row=0,column=1,sticky="ns")

# ------------------------------------------------list box-------------------------------------------------------------------------------#

Listbook=("clean code","Data Structure & Algorithm Classes","BigData","Designthinking","Core programming",

"Full Stack Development","Complete Data Science Program Trending.",

"coding with c", "c++programming","pythontkinter",

"softwer design with uml","programming with c","Rprogramming","SQL database")

def SelectBook(event=""):

value=str(listBox.get(listBox.curselection()))

x=value

if (x=="clean code"):

self.Book\_ID\_var.set("BKID5454")

self.Book\_Name\_var.set("clean code")

d1=datetime.datetime.today()

d2=datetime.timedelta(days=15)

d3=d1+d2

self.Date\_Borrowed\_var.set(d1)

self.Due\_Date\_var.set(d3)

self.Late\_Return\_var.set("Rs.50")

self.Date\_Over\_Due\_var.set("No")

self.ActualPrice\_var.set("Rs.512")

elif (x=="Data Structure & Algorithm Classes"):

self.Book\_ID\_var.set("BKID5485")

self.Book\_Name\_var.set("Data Structure & Algorithm")

d1=datetime.datetime.today()

d2=datetime.timedelta(days=15)

d3=d1+d2

self.Date\_Borrowed\_var.set(d1)

self.Due\_Date\_var.set(d3)

self.Late\_Return\_var.set("Rs.50")

self.Date\_Over\_Due\_var.set("No")

self.ActualPrice\_var.set("Rs.790")

elif (x=="Big Data"):

self.Book\_ID\_var.set("BKID5490")

self.Book\_Name\_var.set("Big Data")

d1=datetime.datetime.today()

d2=datetime.timedelta(days=15)

d3=d1+d2

self.Date\_Borrowed\_var.set(d1)

self.Due\_Date\_var.set(d3)

self.Late\_Return\_var.set("Rs.50")

self.Date\_Over\_Due\_var.set("No")

self.ActualPrice\_var.set("Rs.800")

elif (x=="Design thinking"):

self.Book\_ID\_var.set("BKID5824")

self.Book\_Name\_var.set("Design thinking")

d1=datetime.datetime.today()

d2=datetime.timedelta(days=15)

d3=d1+d2

self.Date\_Borrowed\_var.set(d1)

self.Due\_Date\_var.set(d3)

self.Late\_Return\_var.set("Rs.50")

self.Date\_Over\_Due\_var.set("No")

self.ActualPrice\_var.set("Rs.400")

elif (x=="Core programming"):

self.Book\_ID\_var.set("BKID5047")

self.Book\_Name\_var.set("Core programming")

d1=datetime.datetime.today()

d2=datetime.timedelta(days=15)

d3=d1+d2

self.Date\_Borrowed\_var.set(d1)

self.Due\_Date\_var.set(d3)

self.Late\_Return\_var.set("Rs.50")

self.Date\_Over\_Due\_var.set("No")

self.ActualPrice\_var.set("Rs.700")

elif (x=="Full Stack Development"):

self.Book\_ID\_var.set("BKID6454")

self.Book\_Name\_var.set("Full Stack Development")

d1=datetime.datetime.today()

d2=datetime.timedelta(days=15)

d3=d1+d2

self.Date\_Borrowed\_var.set(d1)

self.Due\_Date\_var.set(d3)

self.Late\_Return\_var.set("Rs.50")

self.Date\_Over\_Due\_var.set("No")

self.ActualPrice\_var.set("Rs.900")

elif (x=="Complete Data Science Program Trending."):

self.Book\_ID\_var.set("BKID4454")

self.Book\_Name\_var.set("Data Science Programing")

d1=datetime.datetime.today()

d2=datetime.timedelta(days=15)

d3=d1+d2

self.Date\_Borrowed\_var.set(d1)

self.Due\_Date\_var.set(d3)

self.Late\_Return\_var.set("Rs.50")

self.Date\_Over\_Due\_var.set("No")

self.ActualPrice\_var.set("Rs.900")

elif (x=="coding with c"):

self.Book\_ID\_var.set("BKID5325")

self.Book\_Name\_var.set("coding with c")

d1=datetime.datetime.today()

d2=datetime.timedelta(days=15)

d3=d1+d2

self.Date\_Borrowed\_var.set(d1)

self.Due\_Date\_var.set(d3)

self.Late\_Return\_var.set("Rs.50")

self.Date\_Over\_Due\_var.set("No")

self.ActualPrice\_var.set("Rs.612")

elif (x=="c++programing"):

self.Book\_ID\_var.set("BKID5454")

self.Book\_Name\_var.set("c++ programming")

d1=datetime.datetime.today()

d2=datetime.timedelta(days=15)

d3=d1+d2

self.Date\_Borrowed\_var.set(d1)

self.Due\_Date\_var.set(d3)

self.Late\_Return\_var.set("Rs.50")

self.Date\_Over\_Due\_var.set("No")

self.ActualPrice\_var.set("Rs.758")

elif (x=="python tkinter"):

self.Book\_ID\_var.set("BKID2454")

self.Book\_Name\_var.set("python tkinter")

d1=datetime.datetime.today()

d2=datetime.timedelta(days=15)

d3=d1+d2

self.Date\_Borrowed\_var.set(d1)

self.Due\_Date\_var.set(d3)

self.Late\_Return\_var.set("Rs.50")

self.Date\_Over\_Due\_var.set("No")

self.ActualPrice\_var.set("Rs.912")

elif (x=="softwer design with uml"):

self.Book\_ID\_var.set("BKID5698")

self.Book\_Name\_var.set("softwer design with uml")

d1=datetime.datetime.today()

d2=datetime.timedelta(days=15)

d3=d1+d2

self.Date\_Borrowed\_var.set(d1)

self.Due\_Date\_var.set(d3)

self.Late\_Return\_var.set("Rs.50")

self.Date\_Over\_Due\_var.set("No")

self.ActualPrice\_var.set("Rs.812")

elif (x=="programming with c"):

self.Book\_ID\_var.set("BKID8454")

self.Book\_Name\_var.set("programming with c")

d1=datetime.datetime.today()

d2=datetime.timedelta(days=15)

d3=d1+d2

self.Date\_Borrowed\_var.set(d1)

self.Due\_Date\_var.set(d3)

self.Late\_Return\_var.set("Rs.50")

self.Date\_Over\_Due\_var.set("No")

self.ActualPrice\_var.set("Rs.712")

elif (x=="R programming"):

self.Book\_ID\_var.set("BKID7454")

self.Book\_Name\_var.set("R programming")

d1=datetime.datetime.today()

d2=datetime.timedelta(days=15)

d3=d1+d2

self.Date\_Borrowed\_var.set(d1)

self.Due\_Date\_var.set(d3)

self.Late\_Return\_var.set("Rs.50")

self.Date\_Over\_Due\_var.set("No")

self.ActualPrice\_var.set("Rs.1012")

elif (x=="SQL database"):

self.Book\_ID\_var.set("BKID7454")

self.Book\_Name\_var.set("SQL database")

d1=datetime.datetime.today()

d2=datetime.timedelta(days=15)

d3=d1+d2

self.Date\_Borrowed\_var.set(d1)

self.Due\_Date\_var.set(d3)

self.Late\_Return\_var.set("Rs.50")

self.Date\_Over\_Due\_var.set("No")

self.ActualPrice\_var.set("Rs.712")

elif (x=="c++ programming"):

self.Book\_ID\_var.set("BKID7400")

self.Book\_Name\_var.set("c++ programming")

d1=datetime.datetime.today()

d2=datetime.timedelta(days=15)

d3=d1+d2

self.Date\_Borrowed\_var.set(d1)

self.Due\_Date\_var.set(d3)

self.Late\_Return\_var.set("Rs.50")

self.Date\_Over\_Due\_var.set("No")

self.ActualPrice\_var.set("Rs.912")

listBox=Listbox(DataFrameRight,font=("times new roman",15,"bold"),width=20,height=13)

listBox.bind("<<ListboxSelect>>",SelectBook)

listBox.grid(row=0,column=0,padx=4)

listScrollbar.config(command=listBox.yview)

for item in Listbook:

listBox.insert(END,item)

#-------------------------------------------- Button frame ------------------------------------------------------------------------ #

framebutton=Frame(self.root,bd=12,relief=RIDGE,padx=20,bg="powder blue")

framebutton.place(x=0,y=500,width=1370,height=70)

btnAddData=Button(framebutton,command=idata,text="add\_data",font=("times new roman",15,"bold"),width=15,bg="blue",fg="white")

btnAddData.grid(row=0,column=0)

btnShowData=Button(framebutton,command=self.showdata,text="Show Data",font=("times new roman",15,"bold"),width=15,bg="blue",fg="white")

btnShowData.grid(row=0,column=2)

btnUpdate=Button(framebutton,command=self.update,text="Update",font=("times new roman",15,"bold"),width=15,bg="blue",fg="white")

btnUpdate.grid(row=0,column=3)

btnDelete=Button(framebutton,command=self.delete,text="Delete",font=("times new roman",15,"bold"),width=15,bg="blue",fg="white")

btnDelete.grid(row=0,column=4)

btnReset=Button(framebutton,command=self.reset,text="Reset",font=("times new roman",15,"bold"),width=15,bg="blue",fg="white")

btnReset.grid(row=0,column=5)

btnExit=Button(framebutton,command=self.iExit,text="Exit",font=("times new roman",15,"bold"),width=20,bg="blue",fg="white")

btnExit.grid(row=0,column=6)

#-------------------------------------------- Information frame ------------------------------------------------------------------- #

Framedetail=Frame(self.root,bd=12,relief=RIDGE,padx=20,bg="powder blue")

Framedetail.place(x=0,y=560,width=1370,height=150)

Table\_frame=Frame(Framedetail,bd=6,relief=RIDGE,bg="powder blue")

Table\_frame.place(x=0,y=2,width=1320,height=130)

xscroll=ttk.Scrollbar(Table\_frame,orient=HORIZONTAL)

yscroll=ttk.Scrollbar(Table\_frame,orient=VERTICAL)

self.library\_table=ttk.Treeview(Table\_frame,column=("membertype","idnumber","department","year","name","mobileno","bookid",

"bookname","dateborrowed","datedue","latereturn","dateoverdue","actualprice"),xscrollcommand=xscroll.set,yscrollcommand=yscroll.set)

xscroll.pack(side=BOTTOM,fill=X)

yscroll.pack(side=RIGHT,fill=Y)

xscroll.config(command=self.library\_table.xview)

yscroll.config(command=self.library\_table.yview)

self.library\_table.heading("membertype",text="Member type")

self.library\_table.heading("idnumber",text="ID\_number")

self.library\_table.heading("department",text="Department")

self.library\_table.heading("year",text="Year")

self.library\_table.heading("name",text="Name")

self.library\_table.heading("mobileno",text="Mobile\_no")

self.library\_table.heading("bookid",text="Book\_ID")

self.library\_table.heading("bookname",text="Book\_Name")

self.library\_table.heading("dateborrowed",text="Date\_Borrowed")

self.library\_table.heading("datedue",text="Date\_Due")

self.library\_table.heading("latereturn",text="Late\_Return")

self.library\_table.heading("dateoverdue",text="Date\_Over\_Due")

self.library\_table.heading("actualprice",text="Actualprice")

self.library\_table["show"]="headings"

self.library\_table.pack(fill=BOTH,expand=1)

self.library\_table.column("membertype",width=100)

self.library\_table.column("idnumber",width=100)

self.library\_table.column("department",width=100)

self.library\_table.column("year",width=100)

self.library\_table.column("name",width=100)

self.library\_table.column("mobileno",width=100)

self.library\_table.column("bookid",width=100)

self.library\_table.column("bookname",width=100)

self.library\_table.column("dateborrowed",width=100)

self.library\_table.column("datedue",width=100)

self.library\_table.column("latereturn",width=100)

self.library\_table.column("dateoverdue",width=100)

self.library\_table.column("actualprice",width=100)

self.fatch\_data()

self.library\_table.bind("<ButtonRelease-1>",self.get\_cursor)

def fatch\_data(self):

conn=mysql.connector.connect(host="localhost",username="root",password="mysql",database="librarymanagementsystem")

my\_cursor=conn.cursor()

my\_cursor.execute("select\*from library")

rows=my\_cursor.fetchall()

if len(rows)!=0:

self.library\_table.delete(\*self.library\_table.get\_children())

for i in rows:

self.library\_table.insert("",END,values=i)

conn.commit()

conn.close()

def get\_cursor(self,event=""):

cursor\_row=self.library\_table.focus()

content=self.library\_table.item(cursor\_row)

row=content['values']

self.Member\_var.set(row[0])

self.ID\_number\_var.set(row[1])

self.Department\_var.set(row[2])

self.Year\_var.set(row[3])

self.Name\_var.set(row[4])

self.Mobile\_no\_var.set(row[5])

self.Book\_ID\_var.set(row[6])

self.Book\_Name\_var.set(row[7])

self.Date\_Borrowed\_var.set(row[8])

self.Due\_Date\_var.set(row[9])

self.Late\_Return\_var.set(row[10])

self.Date\_Over\_Due\_var.set(row[11])

self.ActualPrice\_var.set(row[12])

def showdata(self):

self.textbox.insert(END,"Member\_type\t\t"+self.Member\_var.get()+"\n")

self.textbox.insert(END,"Id number\t\t"+self.ID\_number\_var.get()+"\n")

self.textbox.insert(END,"Department\t\t"+self.Department\_var.get()+"\n")

self.textbox.insert(END,"Year\t\t"+self.Year\_var.get()+"\n")

self.textbox.insert(END,"Name\t\t"+self.Name\_var.get()+"\n")

self.textbox.insert(END,"Mobileno\t\t"+self.Mobile\_no\_var.get()+"\n")

self.textbox.insert(END,"BookId\t\t"+self.Book\_ID\_var.get()+"\n")

self.textbox.insert(END,"BookName\t\t"+self.Book\_Name\_var.get()+"\n")

self.textbox.insert(END,"Datebrrowed\t\t"+self.Date\_Borrowed\_var.get()+"\n")

self.textbox.insert(END,"DueDate\t\t"+self.Due\_Date\_var.get()+"\n")

self.textbox.insert(END,"Latereturn\t\t"+self.Late\_Return\_var.get()+"\n")

self.textbox.insert(END,"DateoverDue\t\t"+self.Date\_Over\_Due\_var.get()+"\n")

self.textbox.insert(END,"Actualprice\t\t"+self.ActualPrice\_var.get()+"\n")

def reset(self):

self.Member\_var.set(""),

self.ID\_number\_var.set(""),

self.Department\_var.set(""),

self.Year\_var.set(""),

self.Name\_var.set(""),

self.Mobile\_no\_var.set(""),

self.Book\_ID\_var.set(""),

self.Book\_Name\_var.set(""),

self.Date\_Borrowed\_var.set(""),

self.Due\_Date\_var.set(""),

self.Late\_Return\_var.set(""),

self.Date\_Over\_Due\_var.set(""),

self.ActualPrice\_var.set("")

self.textbox.delete("1.0",END)

def iExit(self):

iExit=tkinter.messagebox.askyesno("Library Management System","Do you want to Exit")

if iExit>0:

self.root.destroy()

return

def update(self):

conn=mysql.connector.connect(host="localhost",username="root",password="mysql",database="librarymanagementsystem")

my\_cursor=conn.cursor()

my\_cursor.execute("update library set Membertype=%s,Department=%s,Year=%s,Name=%s,Mobileno=%s,BookId=%s,BookName=%s,DateBorrowed=%s,DateDue=%s,LateReturn=%s,DateOverDue=%s,Actualprice=%s where IDnumber=%s",(

self.Member\_var.get(),

self.Department\_var.get(),

self.Year\_var.get(),

self.Name\_var.get(),

self.Mobile\_no\_var.get(),

self.Book\_ID\_var.get(),

self.Book\_Name\_var.get(),

self.Date\_Borrowed\_var.get(),

self.Due\_Date\_var.get(),

self.Late\_Return\_var.get(),

self.Date\_Over\_Due\_var.get(),

self.ActualPrice\_var.get(),

self.ID\_number\_var.get()

))

conn.commit()

self.fatch\_data()

self.reset()

conn.close()

messagebox.showinfo("Sucess","Member has been updated")

def delete(self):

if self.ID\_number\_var.get()=="":

messagebox.showerror("Error","First select the member")

else:

conn=mysql.connector.connect(host="localhost",username="root",password="mysql",database="librarymanagementsystem")

my\_cursor=conn.cursor()

query="delete from library where IDnumber=%s"

value=(self.ID\_number\_var.get())

my\_cursor.execute(query,(value,))

conn.commit()

self.fatch\_data()

self.reset()

conn.close

messagebox.showinfo("Sucess","Member has been Deleted")

if \_\_name\_\_=="\_\_main\_\_":

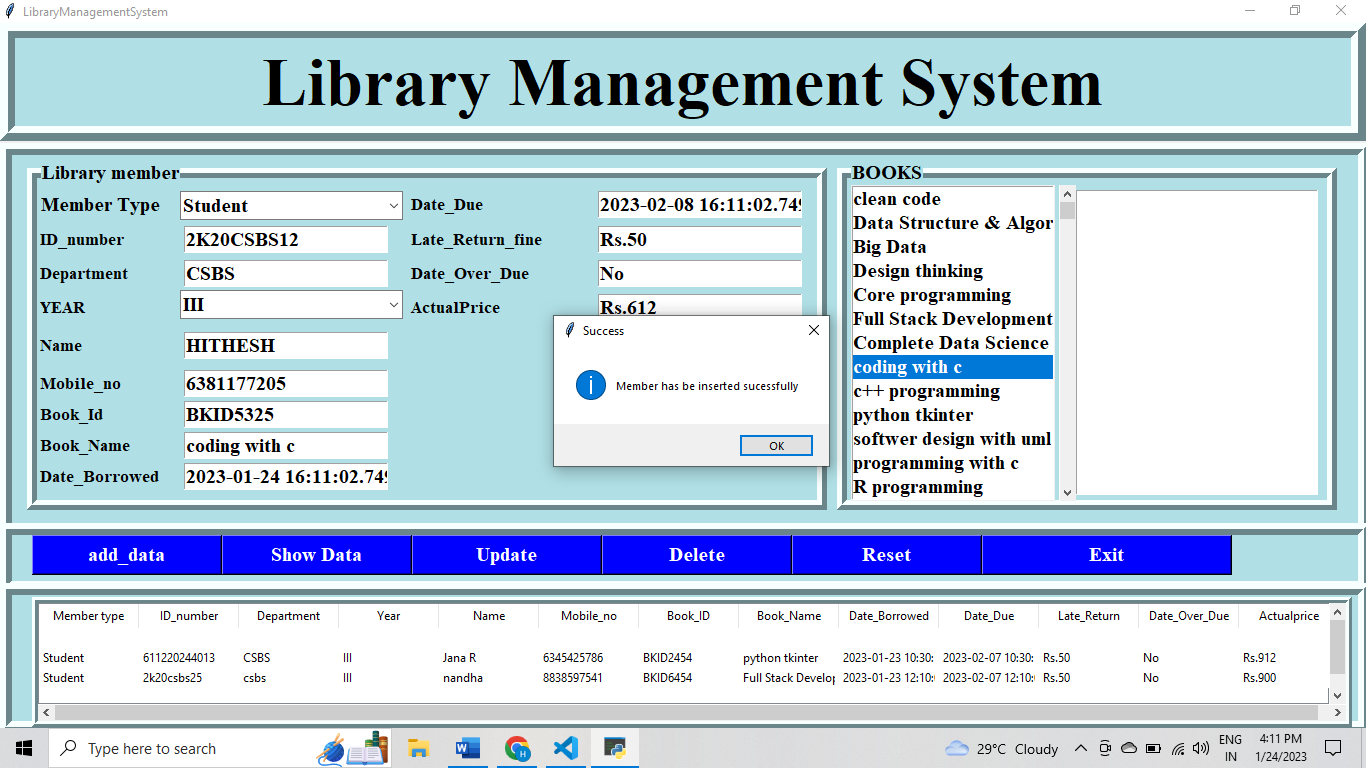
root=Tk()

obj=LibraryManagementSystem(root)

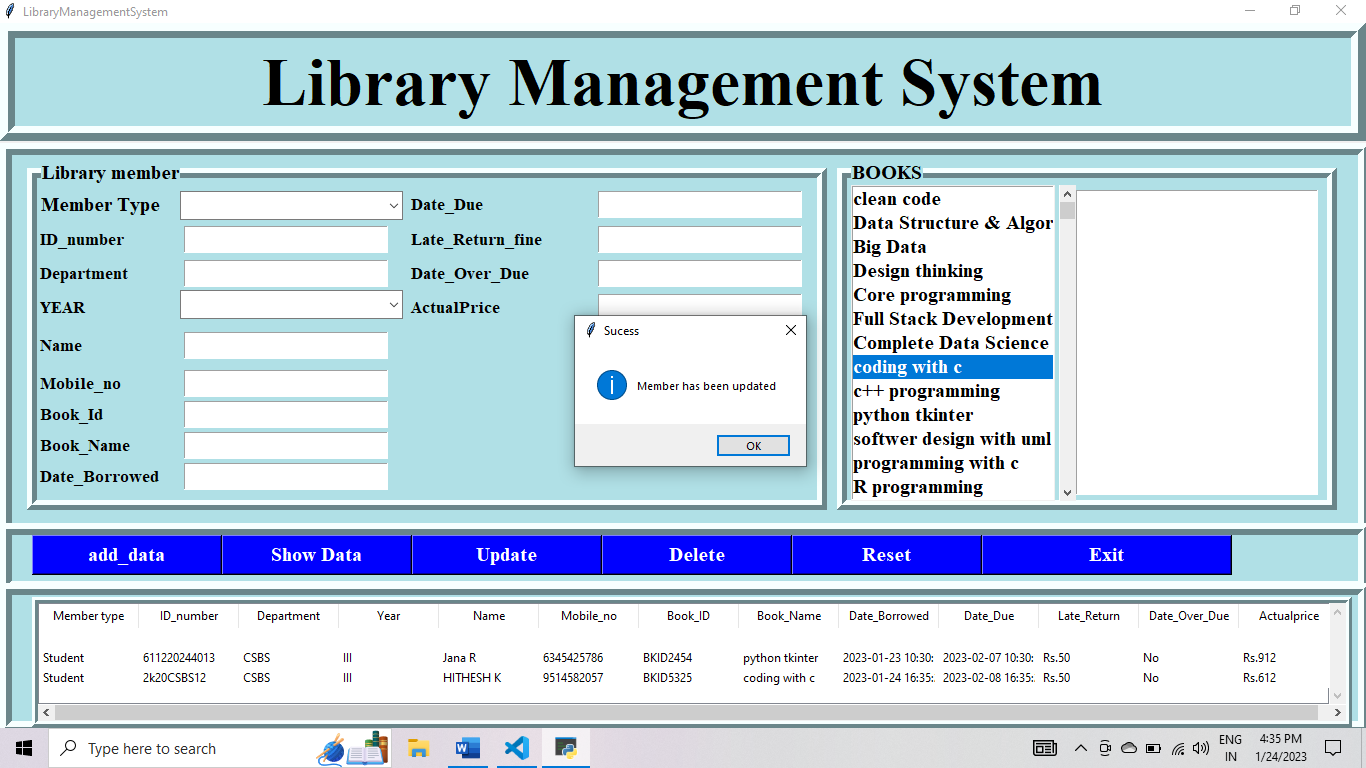
root.mainloop()

**SCREEN SHORTS**

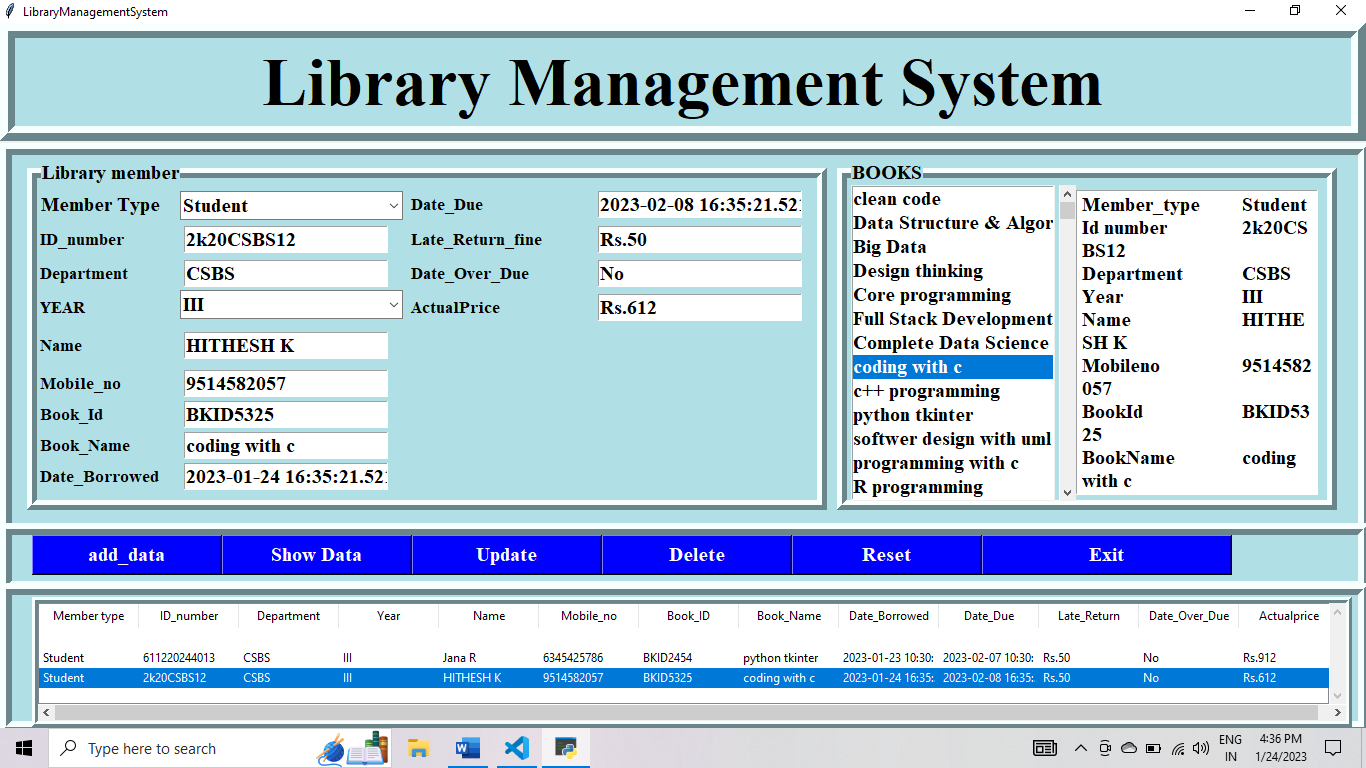
**ADDING NEW MEMBER**

.

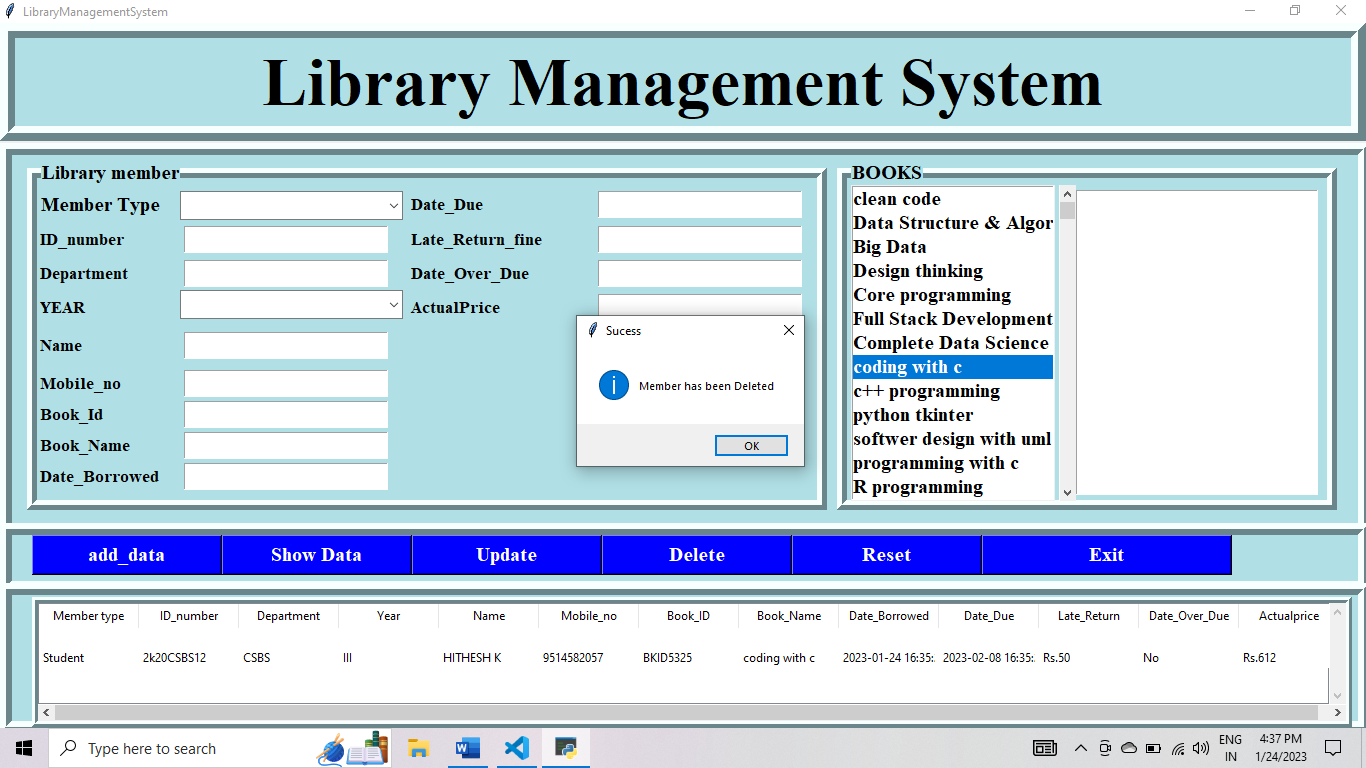
**Updating data of member**



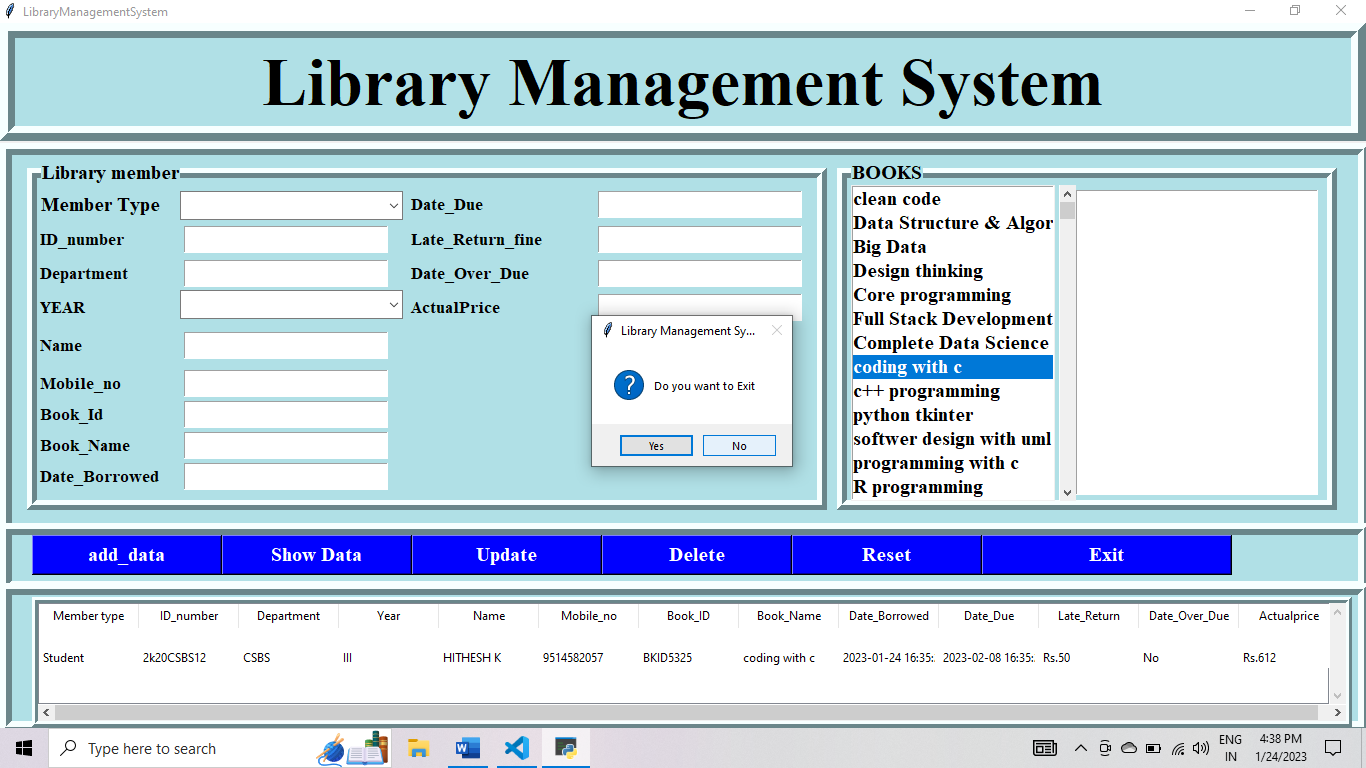
**Showing member data**



**Deleting member from the system**



**Exiting from the system**



**CONCLUSION**

The project library management system was built successfully. The errors in the program which occurred during execution is debugged and the system gives propre output for the give input and it successfully stores the data in the database without any error. Therefore, the software is running properly.

**REFERENCES**

<https://www.w3schools.com/python/>

<https://www.geeksforgeeks.org/python-gui-tkinter/>

<https://www.youtube.com/watch?v=0GHft_BuYhs>

youtube.com/watch?v=xzkTuMUhnDQ&list=WL&index=14

<https://www.youtube.com/watch?v=AIx_P-5rAbY&t=2960s>

<https://www.youtube.com/watch?v=yQSEXcf6s2I&list=PLCC34OHNcOtoC6GglhF3ncJ5rLwQrLGnV>