Batch No.- 3 **Group No.-**

Roll No & Name of Students

70 – Prajwal Korade

71 – Piyush Kothekar

72 – Ayushi Kowe

Title of Course Project

STUDENT MANAGEMENT SYSTEM

Problem Statement: In The earlier system of college to maintain students records was manual paper based system. This system has lots of disadvantages as handling of paper records is very tough as well as time consuming process. AlsoAs the Student data ais also goes on increasing day by day in each school ,colleges ,etc .There is a need of system through which we can easily fetch the details of any student whenever we want .Not only fetch the information but also update ,delete the record of any student easily.

Introduction:

Title of the project is Students Information Management System is defined as anGUI that aims to all the level of management providing information within an organization. To overcome the existing problems and to enhance the process of student's data storage , provide better way to add ,update ,delete and render the data effectively and manage the user's access to the database effectively we can use this simple interface provided by Students Information Management System using python and xampp server.We have used tkinter library in python to create GUI and pymysql connector to connect the database which is created inside the xampp server .

Source code:-

```
from tkinter import *
from tkinter import ttk
import pymysql
from tkinter import messagebox
class Student:
    def __init__(self,root):
        self.root = root
        self.root.title("Student Management System")
        self.root.geometry("1350x700+0+0")
```

```
        title = Label(self.root,text="Student Management
System",bd=10,relief=GROOVE,font=("times new
roman",40,"bold"),bg="Black",fg="Blue")
        title.pack(side = TOP,fill=X)#fill=x to fully cover the head with
title

        #-------All Variables-------
        self.Roll_No_var=StringVar()
        self.Name_var=StringVar()
        self.Email_var=StringVar()
        self.Gender_var=StringVar()
        self.Contact_var=StringVar()
        self.DOB_var=StringVar()

        self.search_by=StringVar()
        self.search_txt=StringVar()

        #-------Manage_Frame--------
        Manage_frame = Frame(self.root,bd =4 ,relief=RIDGE,bg="crimson")
        Manage_frame.place(x=20,y=100,width=450,height=600)
        m_title = Label(Manage_frame,text="Manage
Student",bg="crimson",fg="white",font=("times new roman",20,"bold"),)
        m_title.grid(row=0,columnspan=2,pady=20)

        lbl_roll = Label(Manage_frame,text="Roll
No.",bg="crimson",fg="white",font=("times new roman",20,"bold"))
        lbl_roll.grid(row=1,column=0,pady=10,padx=10,sticky="w ")

        txt_roll =
Entry(Manage_frame,textvariable=self.Roll_No_var,font=("times new
roman",20,"bold"),bd=5,relief=GROOVE)
        txt_roll.grid(row=1,column=1,pady=10,padx=10,sticky="w ")

        lbl_name =
Label(Manage_frame,text="Name",bg="crimson",fg="white",font=("times new
roman",20,"bold"))
        lbl_name.grid(row=2,column=0,pady=10,padx=10,sticky="w ")

        txt_name = Entry(Manage_frame,textvariable=self.Name_var,font=("times
new roman",20,"bold"),bd=5,relief=GROOVE)
        txt_name.grid(row=2,column=1,pady=10,padx=10,sticky="w ")

        lbl_email =
Label(Manage_frame,text="Email",bg="crimson",fg="white",font=("times new
roman",20,"bold"))
        lbl_email.grid(row=3,column=0,pady=10,padx=10,sticky="w ")
```

```
        txt_email =
Entry(Manage_frame,textvariable=self.Email_var,font=("times new
roman",20,"bold"),bd=5,relief=GROOVE)
        txt_email.grid(row=3,column=1,pady=10,padx=10,sticky="w ")

        lbl_gender =
Label(Manage_frame,text="Gender",bg="crimson",fg="white",font=("times new
roman",20,"bold"))
        lbl_gender.grid(row=4,column=0,pady=10,padx=10,sticky="w ")

        combo_gender =
ttk.Combobox(Manage_frame,textvariable=self.Gender_var,font=("times new
roman",19,"bold"),state='readonly ')
        combo_gender['values'] = ('Male','Female')
        combo_gender.grid(row=4,column=1,padx=20,pady=10)

        lbl_contact =
Label(Manage_frame,text="Contact",bg="crimson",fg="white",font=("times new
roman",20,"bold"))
        lbl_contact.grid(row=5,column=0,pady=10,padx=10,sticky="w ")

        txt_contact =
Entry(Manage_frame,textvariable=self.Contact_var,font=("times new
roman",20,"bold"),bd=5,relief=GROOVE)
        txt_contact.grid(row=5,column=1,pady=10,padx=10,sticky="w ")

        lbl_DOB =
Label(Manage_frame,text="DOB",bg="crimson",fg="white",font=("times new
roman",20,"bold"))
        lbl_DOB.grid(row=6,column=0,pady=10,padx=10,sticky="w ")

        txt_DOB = Entry(Manage_frame,textvariable=self.DOB_var,font=("times
new roman",20,"bold"),bd=5,relief=GROOVE)
        txt_DOB.grid(row=6,column=1,pady=10,padx=10,sticky="w ")

        lbl_Add =
Label(Manage_frame,text="Address",bg="crimson",fg="white",font=("times new
roman",20,"bold"))
        lbl_Add.grid(row=7,column=0,pady=10,padx=10,sticky="w ")

        self.txt_Add = Text(Manage_frame,width=40,height=4,font=("",10))
        self.txt_Add.grid(row=7,column=1,pady=10,padx=20,sticky="w ")

        #-----ButtonFrame-----
        btn_frame = Frame(Manage_frame,bd=4,relief=RIDGE,bg="crimson")
        btn_frame.place(x=10,y=540,width=420)
```

```
        Addbtn =
Button(btn_frame,text="ADD",width=10,command=self.add_student).grid(row=0,colu
mn=0,padx=10,pady=10)
        Updabtn =
Button(btn_frame,text="UPDATE",width=10,command=self.update_data
).grid(row=0,column=1,padx=10,pady=10)
        delbtn =
Button(btn_frame,text="DELETE",width=10,command=self.delete_data).grid(row=0,c
olumn=2,padx=10,pady=10)
        Clrbtn =
Button(btn_frame,text="CLEAR",width=10,command=self.clear).grid(row=0,column=3
,padx=10,pady=10)


        #-------Detail_Frame--------
        Detail_frame = Frame(self.root,bd =4 ,relief=RIDGE,bg="crimson")
        Detail_frame.place(x=500,y=100,width=800,height=580)

        lbl_search = Label(Detail_frame,text="Search
By",bg="crimson",fg="white",font=("times new roman",20,"bold"))
        lbl_search.grid(row=0,column=0,pady=10,padx=10,sticky="w ")

        combo_search = ttk.Combobox(Detail_frame,textvariable =
self.search_by,width=10,font=("times new roman",13,"bold"),state='readonly ')
        combo_search['values'] = ('Roll_no','Name','Contact')
        combo_search.grid(row=0,column=1,padx=20,pady=10)

        txt_search = Entry(Detail_frame ,textvariable =
self.search_txt,width=20,font=("times new roman",14
,"bold"),bd=5,relief=GROOVE)
        txt_search.grid(row=0,column=2,pady=10,padx=20,sticky="w ")

        searchbtn =
Button(Detail_frame,text="SEARCH",width=10,pady=5,command=self.search_data).gr
id(row=0,column=3,padx=10,pady=10)
        showallbtn =
Button(Detail_frame,text="SHOWALL",width=10,pady=5,command=self.fetch_data).gr
id(row=0,column=4 ,padx=10,pady=10)

        #--------Table_Frame-----------
        Tbl_frame = Frame(Detail_frame,bd =4 ,relief=RIDGE,bg="crimson")
        Tbl_frame.place(x=10,y=70,width=760,height=495)

        scroll_x = Scrollbar(Tbl_frame,orient=HORIZONTAL)
        scroll_y = Scrollbar(Tbl_frame,orient=VERTICAL)
        self.Student_table =
ttk.Treeview(Tbl_frame,columns=("Roll","Name","Email","Gender","Contact","DOB"
,"Address"),xscrollcommand=scroll_x.set,yscrollcommand=scroll_y.set)
        scroll_x.pack(side=BOTTOM,fill=X)
```

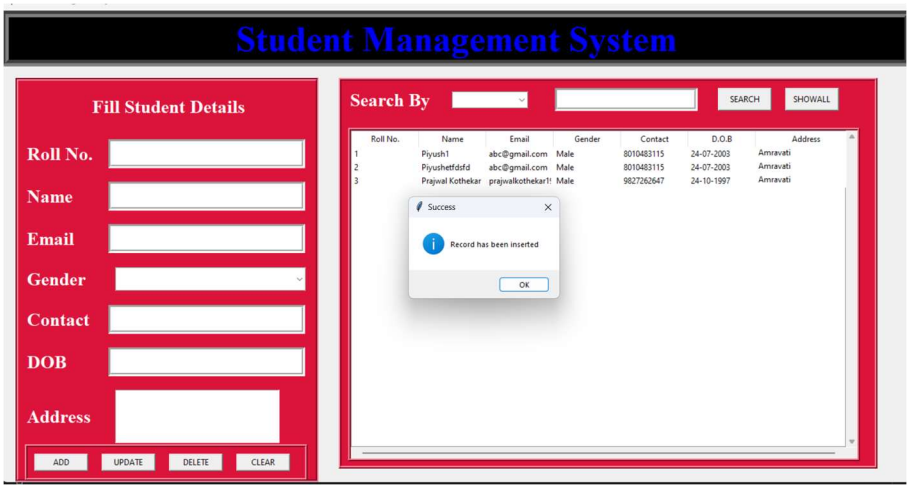
```
        scroll_y.pack(side=RIGHT,fill=Y)
        scroll_x.config(command=self.Student_table.xview)
        scroll_y.config(command=self.Student_table.yview)

        self.Student_table.heading("Roll",text="Roll No.")
        self.Student_table.heading("Name",text="Name")
        self.Student_table.heading("Email",text="Email")
        self.Student_table.heading("Gender",text="Gender")
        self.Student_table.heading("Contact",text="Contact")
        self.Student_table.heading("DOB",text="D.O.B")
        self.Student_table.heading("Address",text="Address")

        self.Student_table['show'] = 'headings'

        self.Student_table.column("Roll",width=100)
        self.Student_table.column("Name",width=100)
        self.Student_table.column("Email",width=100)
        self.Student_table.column("Gender",width=100)
        self.Student_table.column("Contact",width=100)
        self.Student_table.column("DOB",width=100)
        self.Student_table.column("Address",width=150)
        self.Student_table.pack(fill=BOTH,expand=1)

        self.Student_table.bind("<ButtonRelease-1>",self.get_cursor)
        self.fetch_data()

    def add_student(self):
        if self.Roll_No_var.get()=="" or self.Name_var.get()="":
            messagebox.showerror("Error","All fields are required !!!")
        else:
                    con
=  pymysql.connect(host="localhost",user="root",password="Piyu@24July",databas
e="SMS")
                    cur = con.cursor()
                    cur.execute("insert into student
values(%s,%s,%s,%s,%s,%s,%s)",(self.Roll_No_var.get(),self.Name_var.get(),self
.Email_var.get(),self.Gender_var.get(),self.Contact_var.get(),self.DOB_var.get
(),self.txt_Add.get('1.0',END)))
                    con.commit()
                    self.fetch_data()
                    self.clear()
                    con.close()
                    messagebox.showinfo("Success","Record has been inserted")

    def fetch_data(self):
        con
=  pymysql.connect(host="localhost",user="root",password="Piyu@24July",databas
e="SMS")
```

```
        cur = con.cursor()
        cur.execute("select * from student")
        rows = cur.fetchall()
        if len(rows)!=0:
            self.Student_table.delete(*self.Student_table.get_children())
            for row in rows:
                self.Student_table.insert('',END,values=row)
            con.commit()
        con.close()

    def search_data(self):
        con
=  pymysql.connect(host="localhost",user="root",password="Piyu@24July",databas
e="SMS")
        cur = con.cursor()
        cur.execute("select * from student where "+str(self.search_by.get())+"
LIKE '%"+str(self.search_txt.get())+"%'")
        rows = cur.fetchall()
        if len(rows)!=0:
            self.Student_table.delete(*self.Student_table.get_children())
            for row in rows:
                self.Student_table.insert('',END,values=row)
            con.commit()
        con.close()

    def clear(self):
        self.Roll_No_var.set("")
        self.Name_var.set("")
        self.Email_var.set("")
        self.Gender_var.set("")
        self.Contact_var.set("")
        self.DOB_var.set("")
        self.txt_Add.delete('1.0',END) #Delete all the text right from
starting to end

    def get_cursor(self,ev):
        cursor_row = self.Student_table.focus() #fetch the row where cursor
points
        content = self.Student_table.item(cursor_row) #stores the data in
content for that row
        row = content['values'] #return list of values inside that row
        #print(row)
        self.Roll_No_var.set(row[0])
        self.Name_var.set(row[1])
        self.Email_var.set(row[2])
        self.Gender_var.set(row[3])
        self.Contact_var.set(row[4])
        self.DOB_var.set(row[5])
```

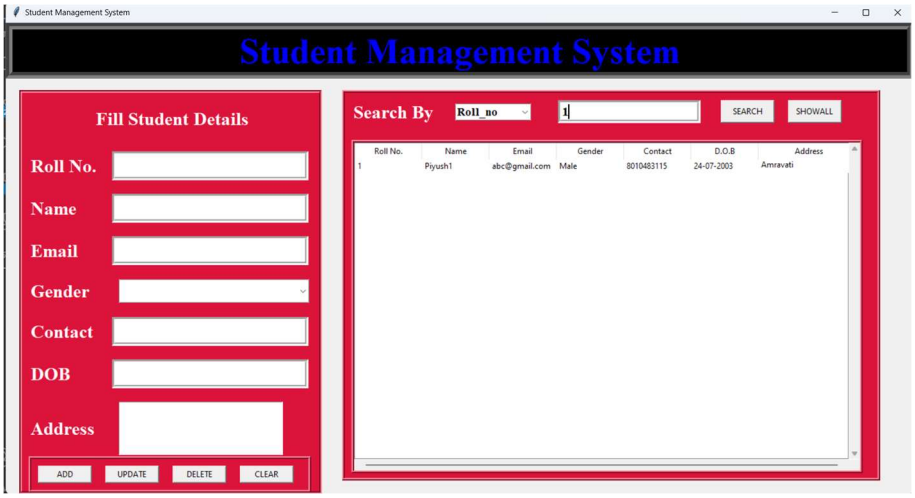
```
        self.txt_Add.delete('1.0',END)
        self.txt_Add.insert(END,row[6])

    def update_data(self):
        con
=  pymysql.connect(host="localhost",user="root",password="Piyu@24July",databas
e="SMS")
        cur = con.cursor()
        cur.execute("update student set
Name=%s,Email=%s,Gender=%s,Contact=%s,DOB=%s,Address=%s where Roll_no=%s
",(self.Name_var.get(),self.Email_var.get(),self.Gender_var.get(),self.Contact
_var.get(),self.DOB_var.get(),self.txt_Add.get('1.0',END),self.Roll_No_var.get
()))
        con.commit()
        self.fetch_data()
        self.clear()
        con.close()

    def delete_data(self):
        con
=  pymysql.connect(host="localhost",user="root",password="Piyu@24July",databas
e="SMS")
        cur = con.cursor()
        cur.execute("delete from student where
Roll_no=%s",self.Roll_No_var.get())
        con.commit()
        con.close()
        self.fetch_data()
        self.clear()

root = Tk()
ob = Student(root)
root.mainloop()
```

Results:-

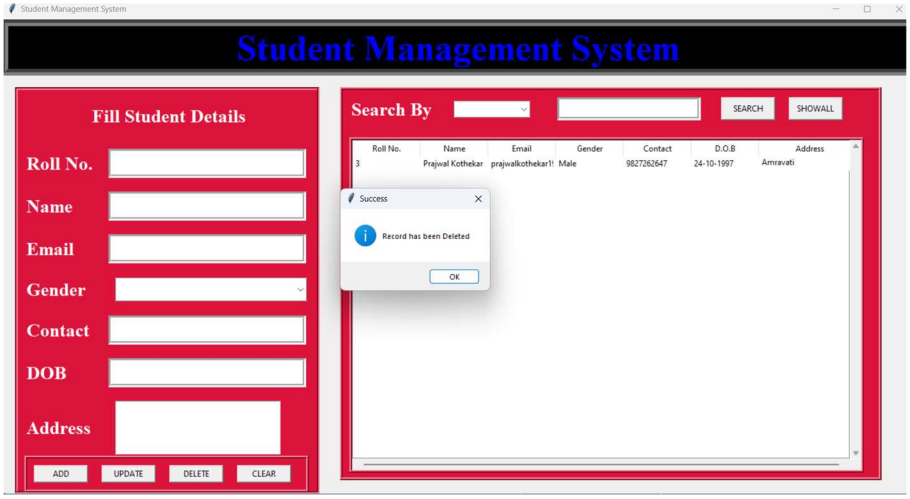
1)Added the details of one student



1)Searched and displayed the details of rollno1 student



3)Deleted the record of one student



Conclusion:

This student information management system used to store the student information easily. It also used to manipulate student information. By this student information management system we can maintain the student data efficiently.The overview of system elaborates the ease of information delivery at the tip of your fingers with precise data and increases the retention rate of student and teaches them how to manage their time efficiently.
We conclude that the present system would definitely help the user by saving time and effort by reducing the processing time.The user satisfaction would be definitely higher when compared to the old manual system

References:

1] Gomathy, C K. (2022). STUDENT INFORMATION MANAGEMENT SYSTEM. INTERANTIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT. 06. 10.55041/IJSREM11816.

2] Mr. Sangamesh K, Mr.Akash Samanekar, Mr.Ningappa T Pujar, 0, Student Management System, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) ICRTT – 2018 (Volume 06 – Issue 15),

3] "STUDENT INFORMATION MANAGEMENT SYSTEM", International Journal of Emerging Technologies and Innovative Research (www.jetir.org | UGC and issn Approved), ISSN:2349-5162, Vol.7, Issue 3, page no. pp2097-2100, March-2020, Available at : http://www.jetir.org/papers/JETIR2003298.pdf

4] https://www.ijedr.org/papers/IJEDR1801002.pdf