A

Report On

**“Student Management System”**

**Under the subject**

**‘Programming with Python[22616]’**

**Submitted by**

|  |  |  |
| --- | --- | --- |
| **Sr. No** | **Enrollment No** | **Name of the student** |
| 1 | 1700100370 | Mr. Patil Pratik Kumar |

**Under the guidance of**

Mr. Emekar S. P.

Department of Computer Engineering

Government Polytechnic, Karad

**1.0 Rationale**

Students are integral part of a education system. Every year students take admission to institute and some leave the college for the further studies. The administrator of college finds it difficult to manage all that information of students. If he/she tries to manage it using files and papers, it is difficult to manage that data. So, we are developing a simple student management system, which will help him/her in managing the data of the students.

We have used python programming language for developing this project. We have used class and function concept for implementing our project. The other programming constructs will be looping statement, conditional statements, etc., We have created GUI using Python.

**2.0 Aim of the project**

This micro-project aims were

1. To analyze working of Student Management System.
2. To address importance of proper Student Management.
3. To study concepts of Python Programming.
4. To develop a Student Management System.

We have achieved all the aims which were decided at the start of the project. We have completed the project with achieving all the aims at the completion.

**3.0 Course Outcomes Achieved**

1. Display message on screen using Python script on IDE.
2. Develop python program to demonstrate use of Operators.
3. Perform operations on data structures in Python.
4. Develop functions for given problem.
5. Handle Exception.

**4.0 Literature Review**

|  |  |  |
| --- | --- | --- |
| **Author (Publication)** | **Contribution** | **Conclusion** |
| AradhnaSahu  (ResearchGate) | This paper consisted of process of downloading python packages. | We used this paper to download pymysql package. |
| Devaki Ramaswamy (ResearchGate) | This paper consisted of the information about python programming concepts. | We used this paper to understand the concepts of python programming. |
| K. R. Srinath (IRJET) | This paper explains the features of python programming. | We used this paper to understand features of python programming. |
| Kalyan Adawadkar (IJAERD) | This paper explains database connectivity using python. | We used this paper to understand database connectivity using python. |
| Manual Hernandez (Academia) | This research paper contained information about the implementing GUI components using python. | We used this paper for designing GUI using python. |
| Vicky Victor Lobo (Academia) | This paper explains the working of Student Management System. | We used this paper to understand working of Student Management System. |

**Table 1- Literature Review**

**5.0 Actual procedure followed**

Actual procedure followed during the project is mentioned in the table below with the planned start date and completed finish date.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Details of Activity** | **Planned Start Date** | **Completed Finish Date** | **Name of responsible Team Members** |
| 1. | Collecting information about Student Management System | 16/12/2019 | 27/12/2019 | 1. Patil Pratik  2. PawarViwek |
| 2. | Collecting the information about concepts of Python | 28/12/2019 | 10/01/2019 | 1. Pratik Patil  2. Kare Abhilash  3. PawarViwek |
| 3. | Developing the rough design of the system | 16/01/2019 | 22/01/2019 | 1. Patil Pratik  2. PawarViwek |
| 4. | Actual Coding | 23/01/2019 | 31/01/2019 | 1. PawarViwek  2. Kare Abhilash |
| 5. | Testing | 1/02/2019 | 11/02/2019 | 1. Pratik Patil  2. Kare Abhilash  3. PawarViwek |
| 6. | Preparing final project report | 12/02/2019 | 22/02/2019 | 1. Pratik Patil  2. Kare Abhilash  3. PawarViwek |

**Table 2- Actual procedure followed**

**Code**

from tkinter import \*

from tkinter import ttk

from tkinter import messagebox

import pymysql

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 ManagementSystem",bd=10,relief=GROOVE,font=("times new roman",40,"bold"),bg="yellow",fg="red")

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

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=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 Students",bg="crimson",fg="white",font=("times new roman",30,"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=20,sticky="w")

txt\_Roll=Entry(Manage\_Frame,font=("times new roman",15,"bold"),textvariable=self.Roll\_No\_var,bd=5,relief=GROOVE)

txt\_Roll.grid(row=1,column=1,pady=10,padx=20,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=20,sticky="w")

txt\_name=Entry(Manage\_Frame,font=("times new roman",15,"bold"),textvariable=self.name\_var,bd=5,relief=GROOVE)

txt\_name.grid(row=2,column=1,pady=10,padx=20,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=20,sticky="w")

txt\_Email=Entry(Manage\_Frame,font=("times new roman",15,"bold"),textvariable=self.email\_var,bd=5,relief=GROOVE)

txt\_Email.grid(row=3,column=1,pady=10,padx=20,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=20,sticky="w")

combo\_gender=ttk.Combobox(Manage\_Frame,font=("times new roman",13,"bold"),textvariable=self.gender\_var,state="readonly")

combo\_gender['values']=('Male','Female','Other')

combo\_gender.grid(row=4,column=1,pady=10,padx=20)

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=20,sticky="w")

txt\_Contact=Entry(Manage\_Frame,font=("times new roman",15,"bold"),textvariable=self.contact\_var,bd=5,relief=GROOVE)

txt\_Contact.grid(row=5,column=1,pady=10,padx=20,sticky="w")

lbl\_dob=Label(Manage\_Frame,text="D. O. B.",bg="crimson",fg="white",font=("times new roman",20,"bold"))

lbl\_dob.grid(row=6,column=0,pady=10,padx=20,sticky="w")

txt\_Dob=Entry(Manage\_Frame,font=("times new roman",15,"bold"),textvariable=self.dob\_var,bd=5,relief=GROOVE)

txt\_Dob.grid(row=6,column=1,pady=10,padx=20,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=20,sticky="w")

self.txt\_Address=Text(Manage\_Frame,width=30,height=4,font=("times new roman",10,"bold"))

self.txt\_Address.grid(row=7,column=1,pady=10,padx=20,sticky="w")

btn\_Frame=Frame(Manage\_Frame,bd=4,relief=RIDGE,bg="crimson")

btn\_Frame.place(x=15,y=530,width=420)

addbtn=Button(btn\_Frame,text="Add",width=10,command=self.add\_student).grid(row=0,column=0,padx=10,pady=10)

updatebtn=Button(btn\_Frame,text="Update",width=10,command=self.update\_data).grid(row=0,column=1,padx=10,pady=10)

deletebtn=Button(btn\_Frame,text="Delete",width=10,command=self.delete\_data).grid(row=0,column=2,padx=10,pady=10)

clearbtn=Button(btn\_Frame,text="Clear",width=10,command=self.clear).grid(row=0,column=3,padx=10,pady=10)

Detail\_Frame=Frame(self.root,bd=4,relief=RIDGE,bg="crimson")

Detail\_Frame.place(x=500,y=100,width=800,height=600)

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=20,sticky="w") combo\_search=ttk.Combobox(Detail\_Frame,width=10,textvariable=self.search\_by,font=("times new roman",13,"bold"),state="readonly")

combo\_search['values']=('Roll\_no','Name','Contact')

combo\_search.grid(row=0,column=1,pady=10,padx=20)

txt\_search=Entry(Detail\_Frame,width=20,textvariable=self.search\_txt,font=("times new roman",10,"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).grid(row=0,column=3,padx=10,pady=10)

showbtn=Button(Detail\_Frame,text="Show All",width=10,pady=5,command=self.fetch\_data).grid(row=0,column=4,padx=10,pady=10)

Table\_Frame=Frame(Detail\_Frame,bd=4,relief=RIDGE,bg="crimson")

Table\_Frame.place(x=10,y=70,width=760,height=500)

scroll\_x=Scrollbar(Table\_Frame,orient=HORIZONTAL)

scroll\_y=Scrollbar(Table\_Frame,orient=VERTICAL)

self.Student\_table=ttk.Treeview(Table\_Frame,columns=("roll\_no","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\_no",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="DOB")

self.Student\_table.heading("Address",text="Address")

self.Student\_table['show']='headings'

self.Student\_table.column("roll\_no",width=80)

self.Student\_table.column("name",width=110)

self.Student\_table.column("email",width=110)

self.Student\_table.column("gender",width=110)

self.Student\_table.column("contact",width=110)

self.Student\_table.column("dob",width=110)

self.Student\_table.column("Address",width=130)

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()=="" or self.email\_var.get()=="" or self.gender\_var.get()=="" or self.contact\_var.get()=="" or self.dob\_var.get()=="":

messagebox.showerror("Error","All Fields are Required!!")

else:

con=pymysql.connect(host="localhost",user="root",port=3306,password="root",database="stm",cursorclass=pymysql.cursors.DictCursor)

try:

with con.cursor() as cursor:

sqlq="INSERT INTO student values(%s,%s,%s,%s,%s,%s,%s)"

cursor.execute(sqlq,(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\_Address.get('1.0',END)))

con.commit()

self.fetch\_data()

self.clear()

finally:

con.close()

messagebox.showinfo("Success","Record has been inserted")

def fetch\_data(self):

con=pymysql.connect(host="localhost",user="root",password="root",database="stm")

cursor=con.cursor()

cursor.execute("select \* from student")

rows=cursor.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\_Address.delete("1.0",END)

def get\_cursor(self,ev):

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

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

row=content['values']

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\_Address.delete("1.0",END)

self.txt\_Address.insert(END,row[6])

def update\_data(self):

if self.Roll\_No\_var.get()=="" or self.name\_var.get()=="" or self.email\_var.get()=="" or self.gender\_var.get()=="" or self.contact\_var.get()=="" or self.dob\_var.get()=="":

messagebox.showerror("Error","All Fields are Required!!")

else:

con=pymysql.connect(host="localhost",user="root",password="root",database="stm")

cursor=con.cursor()

cursor.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\_Address.get('1.0',END),self.Roll\_No\_var.get()))

con.commit()

self.fetch\_data()

self.clear()

con.close()

messagebox.showinfo("Success","Record has been updated")

def delete\_data(self):

if self.Roll\_No\_var.get()=="":

messagebox.showerror("Error","Roll No is required!!")

else:

con=pymysql.connect(host="localhost",user="root",password="root",database="stm")

cursor=con.cursor()

cursor.execute("delete from student where roll\_no=%s",self.Roll\_No\_var.get())

con.commit()

con.close()

self.fetch\_data()

self.clear()

messagebox.showinfo("Success","Record has been deleted")

def search\_data(self):

if self.search\_txt.get()=="" or self.search\_by.get()=="":

messagebox.showerror("Error","Search Fields are Required!!")

else:

con=pymysql.connect(host="localhost",user="root",password="root",database="stm")

cursor=con.cursor()

cursor.execute("select \* from student where "+str(self.search\_by.get())+" LIKE '%"+str(self.search\_txt.get())+"%'")

rows=cursor.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()

root=Tk()

ob=Student(root)

root.mainloop()

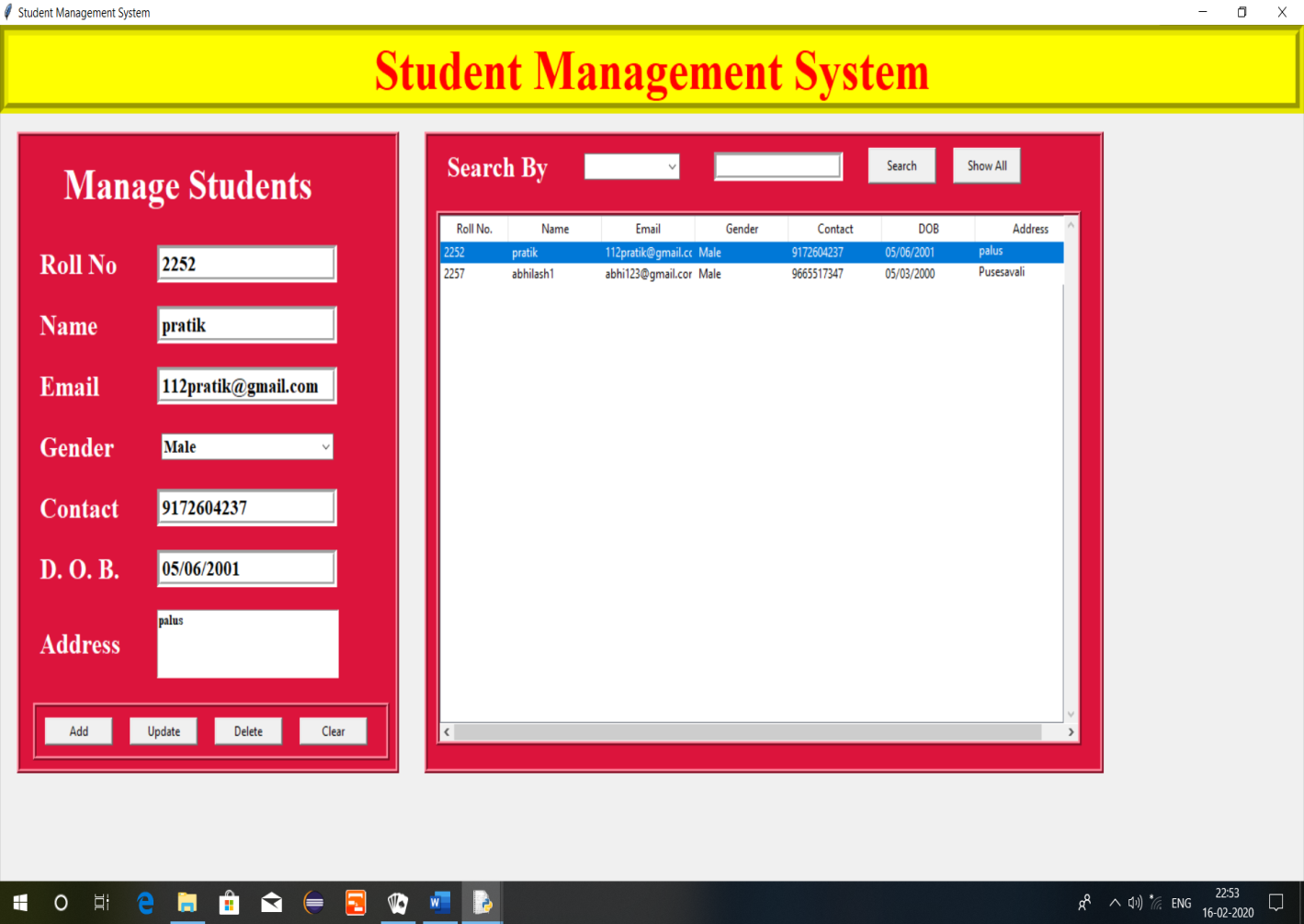
**6.0 Actual Resources Required**

The resources used during the completion of project are mentioned in the below table:

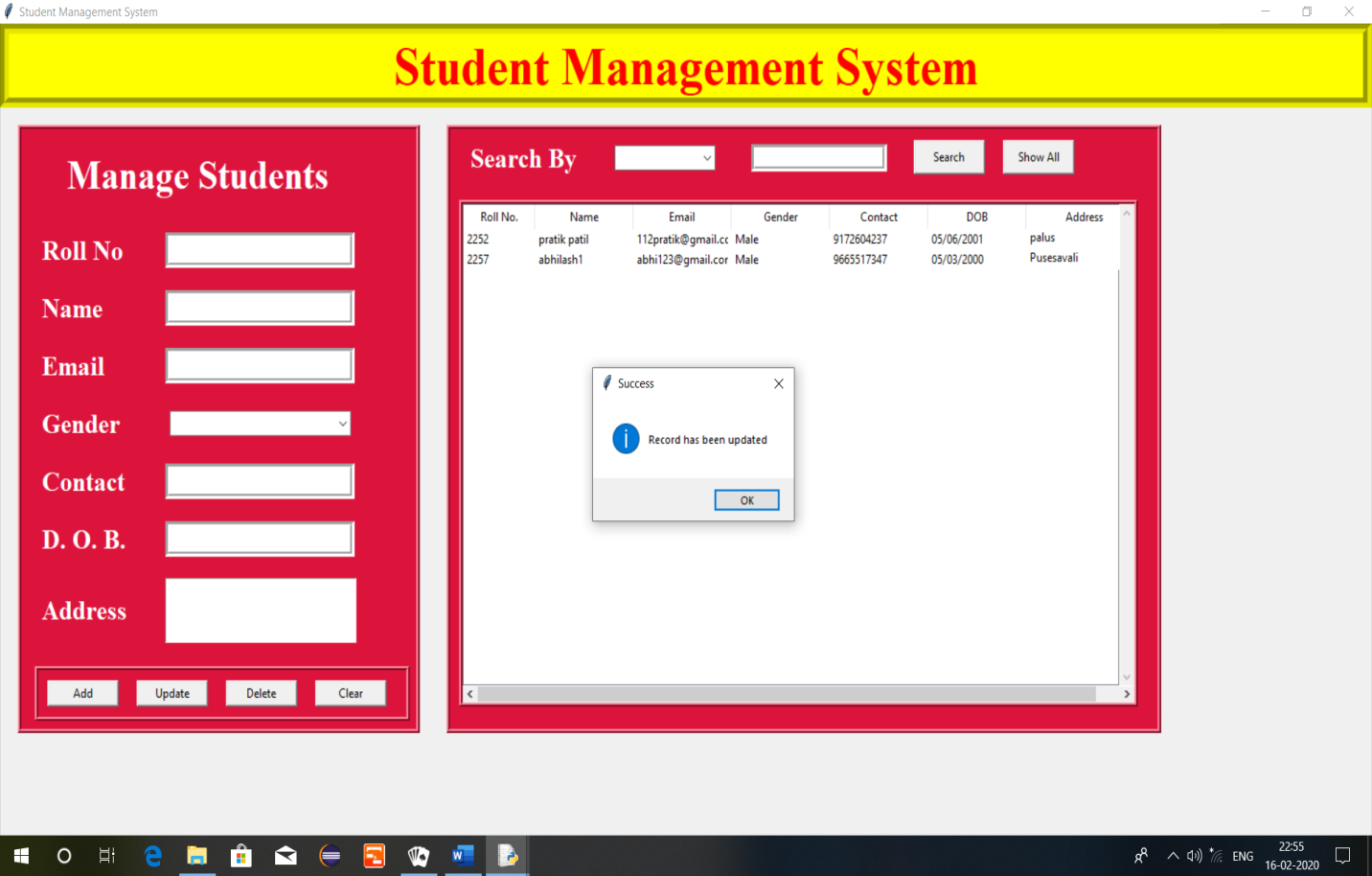
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **Name of resources material** | **Specifications** | **Quantity** | **Remarks** |
| 1. | YouTube | MP4-file format, 640 x 360 pixels | 1 |  |
| 2. | Microsoft Word | 2010 version | 1 |  |
| 3. | IDLE | Version 3.8, 64-bit | 1 |  |
| 4. | Laptop | Dell, RAM 8 GB, Harddisk 1 TB, i3 processor | 1 |  |

**Table 3- Resources required**

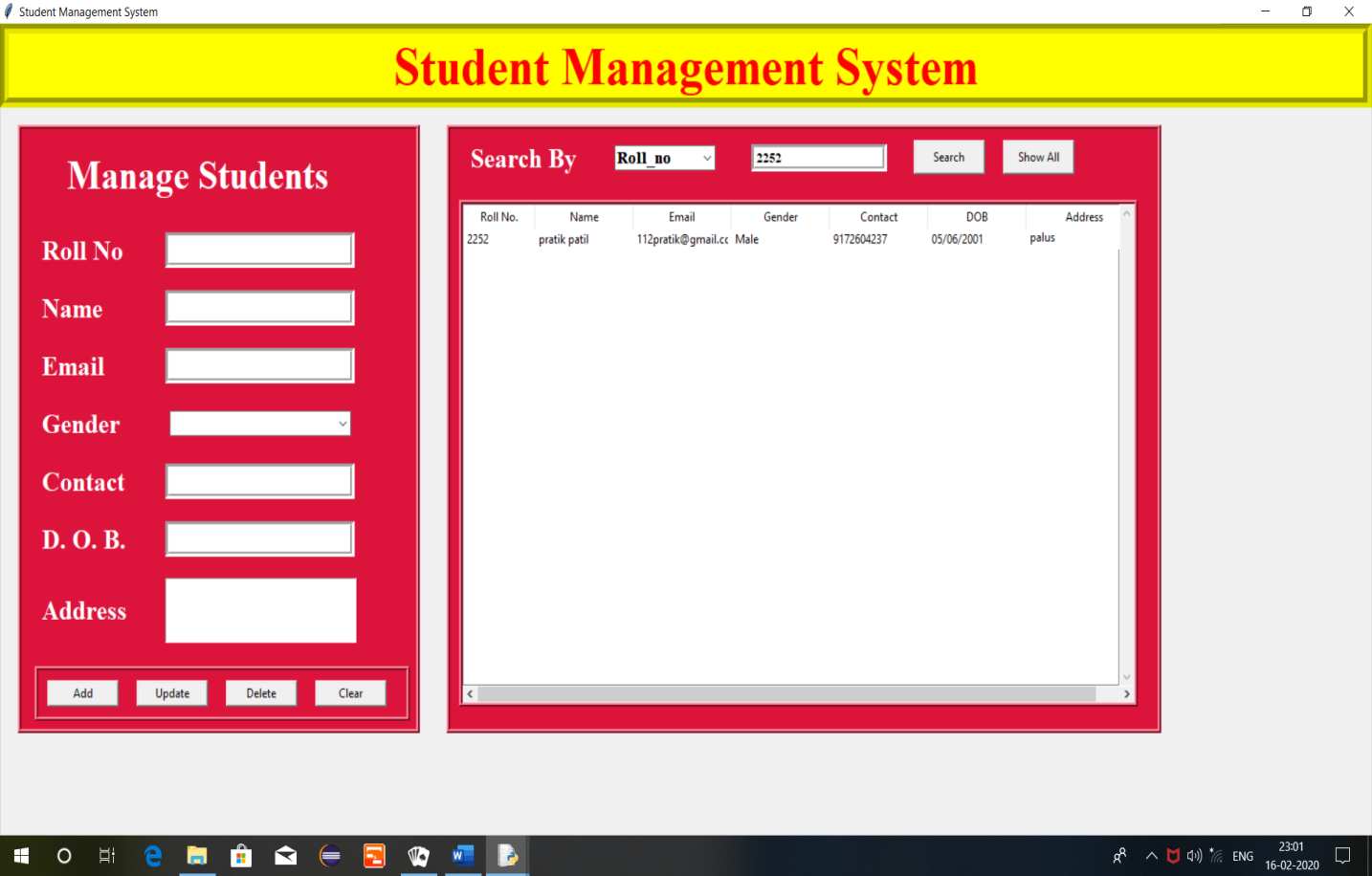
**7.0 Outputs of the Micro-project**



**Fig 1. Displaying selected data in data fields**



**Fig 2. After Updating the data successfully**



**Fig 3. After Searching the data successfully**

**8.0 Skill Developed/ learning out of this Micro-Project**

We learnt,

1. Analyzing real world applications of Python Programming Language.
2. Importance of Student Management System.
3. Developing Student Management System using Python Programming Language.
4. Efficient communication skills.
5. Working as a team.
6. Developing leadership qualities.

**9.0 Applications of the Project**

1. This project can be used in the understanding the importance of Python Programming Language.
2. The project can be also used to understand different concepts of Python Programming Language.
3. The project can be used by colleges to manage student data.

**Subject Teacher**

**Mr. Emekar S. P.**