DELHI PUBLIC SCHOOL HARNI

ACADEMIC SESSION 2021-22

Project Title
Computer Institute Management System

Name of the Student: Prashant Srivastava

Class: XII-A

Roll No:

CERTIFICATE

This is to certify that Prashant Srivastava has successfully completed his Project titled **"Computer Institute Management System"** for the Class XII AISSCE Practical Examination,CBSE for the session 2021-2022. The aforesaid Project Work has been submitted to the Informatics Practices Department of Delhi Public School Harni.

Internal Examiner **External Examiner**

School Stamp

DECLARATION

This is to declare that, to the best of my knowledge, this project is a bona fide work of Prashant Srivastava.

He has worked sincerely on the Project titled **"Computer Institute Management System"** under my supervision and guidance in the School Computer Laboratory and otherwise.

I hereby declare that the aforesaid mentioned work is an authentic work of Prashant Srivastava.

Informatics Practices

Teacher Signature:

ACKNOWLEDGEMENT

I would like to sincerely thank teacher name, our Informatics Practices Teacher Mrs. Preeti Jha for providing me valuable information and guiding me with the relevant topics which facilitated smooth completion of my project.

I would also like to thank our Principal Mrs. Deepali Sekhon for her priceless guidance, constant encouragement constructive comments, which has sustained my efforts during all stages of this project, and my Parents for being a source of encouragement for this project.

INDEX

Purpose of the project:

An Institute faces various difficulties in managing student and staff member's records along with its various attributes associated with this system. They have to maintain various records manually which involves making attendance sheets, making exam results, making payment sheets and defaulter lists and many more. They have to check manually for each and every activity going inside particular institutions. To overcome this problem a computer based Computer Institute Management System is required.

To develop a Computer Institute Management System that will overlook the activities going inside the particular institutions without manual processing. All information should be updated automatically by using the information stored in the database by providing a GUI interface to the end user. The main motive behind this Computer Institute Management System project is to develop a system which will able to handle the overall tasks going inside the institutions without much effort.

Current Computer Institute Management System is not able to maintain dynamic information and not able to keep records of that particular event. To maintain all these records they have to use old process of record keeping system that is by using files and papers. This information can be misused or may include fault entry which will not able to provide correct information. If any error occurs then manual searching and updating process required to correct that particular information.

System Implementation:

Software Used:

- Python as front end
- Sqlite as back end

Software required to run the program:

- Windows 7
- SQLite Server

Hardware required to run the program:

- Processor – Intel Core processor above Pentium IV
- Ram – 512 MB
- Hard Disk – 2 GB
- Monitor.
- Keyboard.

The Document File is made with:

- Microsoft Word 2010
- Microsoft Word 2019

Software Coding- Front-End (Python Spyder):

This Section explores the coding for each frame built in this Software Project

By implementing concepts of Python.Mysqlite3 is the back end software.

(a) Admin login-

The coding is as follows:-

//Importing the required libraries

```
from tkinter import *

from tkinter import ttk

import sqlite3

import tkinter.messagebox

from  datetime import date

from tkinter import filedialog

from tkinter import Text,Tk
```

//Adding code for login button

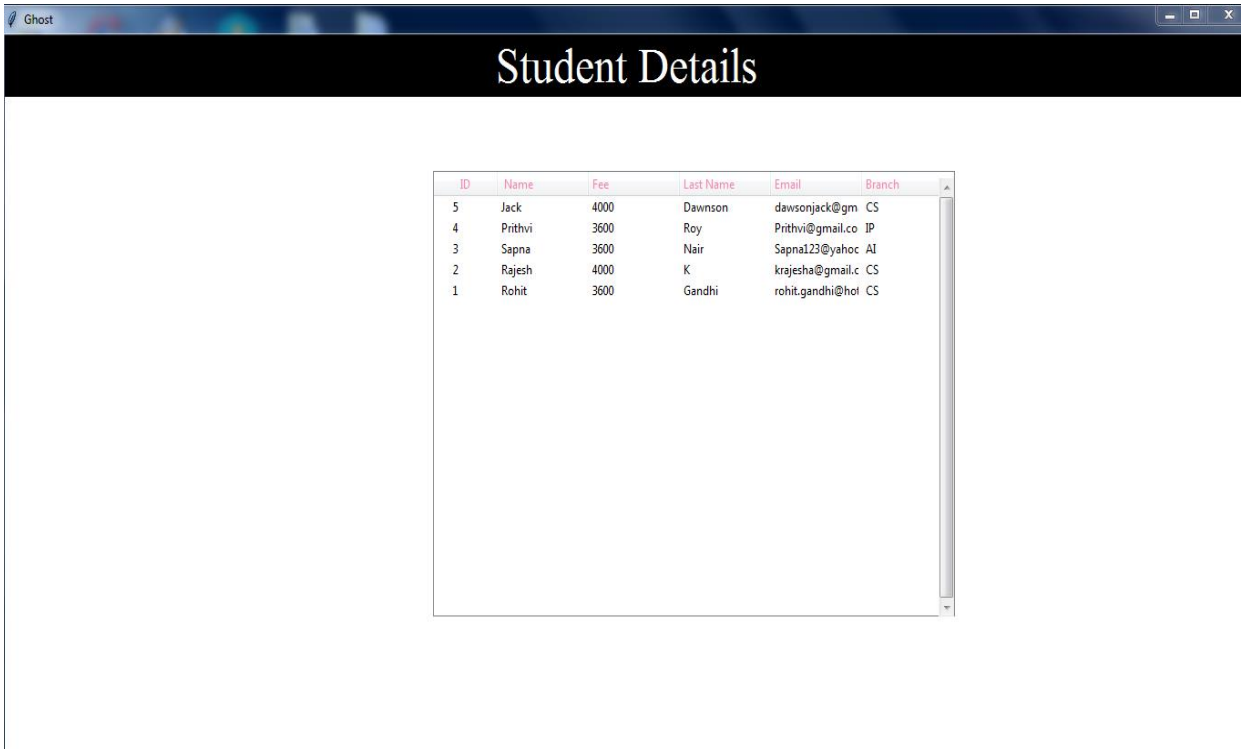
```
today=date.today()

print ('Software is runing.....')

firstw=Tk()

firstw.title("Ghost")

firstw.geometry("1600x1000+0+0")

flash=Label(text="SIS Institute For Computers",font=("Amasis MT
Pro",30),bg="Red",fg="Black")

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

uzer1=Label(text="Username",font=("arial",23))

uzer1.place(x=610,y=120)

uzer=Entry(width=15,bd=4,font=("arial",20))

uzer.place(x=570,y=200)

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

uzer2=Label(text="Password",font=("arial",23))

uzer2.place(x=610,y=280)

uzer3=Entry(width=17,show="*",bd=5,font=("arial",20))

uzer3.place(x=570,y=360)
```

(b) Student Details Window-

```
//Adding code for Registration window

def astral():
```

```
    global astralw
    astralw=Tk()
    astralw.title("Ghost")
    astralw.geometry("1600x1000+0+0")
    def dis4():
        astralw.destroy()
        groot()
    def stu():
        stu1=Tk()
        stu1.title("Student Details")
    def stuid():
        rot = Tk()
        rot.title("Ghost")
        rot.geometry("1600x1000+0+0")
        maple = Label(rot, text="Student Details", font=("times new roman",
35), bg="black",fg="white")
        maple.pack(side=TOP, fill=X)
        pudding1 = ttk.Treeview(rot,height=20,
columns=('name','sur','fee','email','branch'), selectmode="extended")
        pudding1.heading('#0', text='ID', anchor=CENTER)
        pudding1.heading('#1', text=' Name', anchor=W)
        pudding1.heading('#3', text="Last Name", anchor=W)
        pudding1.heading('#2', text='Fee', anchor=W)
        pudding1.heading('#4', text='Email', anchor=W)
        pudding1.heading('#5', text='Branch', anchor=W)
```

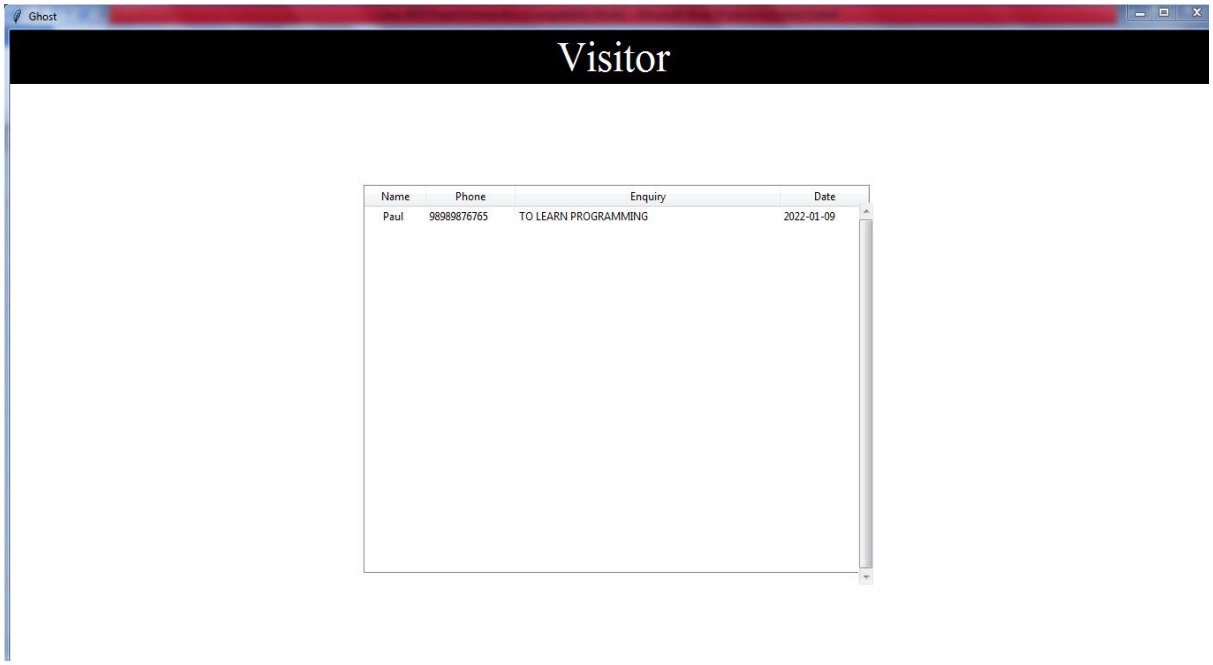
```
    pudding1.column('#1', stretch=YES, minwidth=50, width=100)
    pudding1.column('#3', stretch=YES, minwidth=50, width=100)
    pudding1.column('#2', stretch=YES, minwidth=50, width=100)
    pudding1.column('#0', stretch=YES, minwidth=50, width=70)
    pudding1.column('#4', stretch=YES, minwidth=50, width=100)
    pudding1.column('#5', stretch=YES, minwidth=50, width=100)
    pudding1.place(x=470, y=130)
    ttk.Style().configure("Treeview", background="black",
foreground="coral1")
    ttk.Style().configure("Treeview.Heading", background="blue",
foreground="palevioletRed1")
    rot.configure(background='white')


    vsb=ttk.Scrollbar(rot, orient="vertical",command=pudding1.yview)
    vsb.place(x=1025,y=137,height=400+20)
    pudding1.configure(yscrollcommand=vsb.set)


    conn = sqlite3.connect("IP_Project.db")
    with conn:
      cur = conn.cursor()
      cur.execute('SELECT id ,name, fee ,  sur,email,branch FROM dues ')
      for row1 in cur.fetchall():
        pudding1.insert('', 0, text=row1[0], values=(row1[1]
,row1[2],row1[3],row1[4],row1[5]))
```



(c)View Enquiry Window

```
def ven2():

    rt = Tk()

    rt.title("Ghost")

    rt.geometry("1600x1000+0+0")

    maple =Label(rt, text="Visitor", font=("times new roman", 35), bg="black",fg="white")

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

    pudding1 = ttk.Treeview(rt,height=20 , columns=('Enquiry', 'Date', 'Phone'), selectmode="extended")

    pudding1.heading('#0', text='Name', anchor=CENTER)

    pudding1.heading('#1', text='Phone', anchor=CENTER)

    pudding1.heading('#2', text='Enquiry', anchor=CENTER)

    pudding1.heading('#3', text="Date", anchor=CENTER)
```

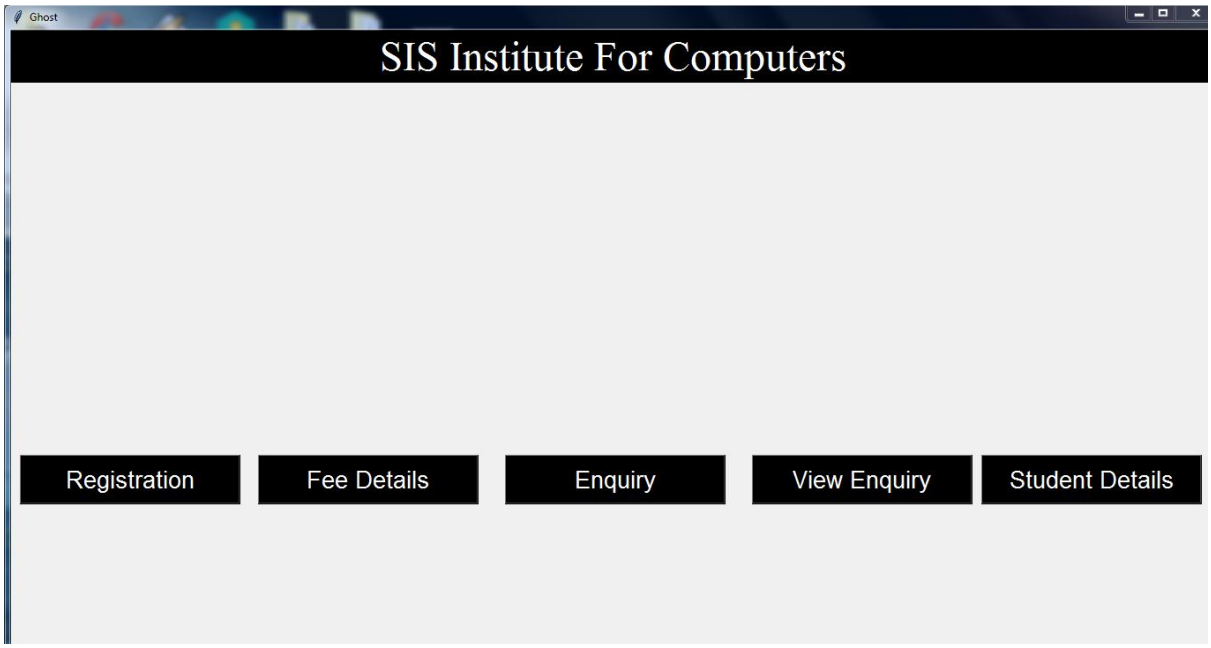
```
    pudding1.column('#1', stretch=YES, minwidth=50, width=100)
    pudding1.column('#3', stretch=YES, minwidth=50, width=100)
    pudding1.column('#2', stretch=YES, minwidth=50, width=300)
    pudding1.column('#0', stretch=YES, minwidth=50, width=70)
    vsb = ttk.Scrollbar(rt, orient="vertical", command=pudding1.yview)
    vsb.place(x=960, y=190, height=400 + 20)
    pudding1.configure(yscrollcommand=vsb.set)
    pudding1.place(x=400, y=170)
    ttk.Style().configure("Treeview", background="#383838",
foreground="coral1")
    ttk.Style().configure("Treeview.heading", background="blue",
foreground="palevioletRed1")
    rt.configure(background="white")
    conn = sqlite3.connect("IP_Project.db")
    with conn:
        cur = conn.cursor()
        cur.execute('SELECT * FROM ex')
        for row in cur.fetchall():
            pudding1.insert('', 0, text=row[0], values=(row[1], row[2],
row[3]))
```



(d) Main Page-

```
def dis5():
    astralw.destroy()
    window()
  maple= Label(astralw,text="SIS Institute For Computers", font=("times new roman", 35), bg="black",fg="white")
  maple.pack(side=TOP, fill=X)
  bttu = Button(astralw,width=15, font=("arial", 20), text="Registration", bg="black",fg="white", command=dis4)
  bttu.place(x=10, y=480)
  eq = Button(astralw, width=15, font=("arial", 20), text="Fee Details", bg="black",fg="white",command=dis5)
  eq.place(x=280, y=480)
  fee_details = Button(astralw, width=15, font=("arial", 20), text="Enquiry", bg="black",fg="white",command=eq1)
```

```
    fee_details.place(x=560, y=480)
    ven= Button(astralw, width=15, font=("arial", 20), text="View Enquiry",
bg="black",fg="white",command=ven2)
    ven.place(x=840, y=480)
    ven1 = Button(astralw, width=15, font=("arial", 20), text="Student
Details", bg="black",fg="white",command=stuid)
    ven1.place(x=1100, y=480)
```




```
def dis():
    firstw.destroy()
def lobby():
    if uzer.get()=="admin" and uzer3.get()=="12345":
        astral()
        dis()
else:
        t = tkinter.messagebox.showinfo("Invalid Username Or Password ",
"You Have Entered Invalid Username Or Password  ")
```

```
    uzer.delete(0,END)
    uzer3.delete(0,END)
```

(e) Registration Window

```
def groot():
    groot=Tk()
    groot.geometry("1600x1000+0+0")
    groot.title("Ghost")
    global lame1
    global lame2
    global lame3
    global lame4
    global lame5
    global box
    global name
    global radio1
    global radio2
    name = StringVar()
    global sur
    sur = StringVar()
    global munch
    munch = IntVar()
    global var1
    var1 = IntVar()
```

```
global var2
var2 = IntVar()
global branch
branch = StringVar()
global rollno
rollno = StringVar()
global email
email = StringVar()
global course
course = StringVar()
global python
python = IntVar()
global java
java = IntVar()
global c
c = IntVar()
global d
d = IntVar()
global calculate
calculate = StringVar()
id = IntVar()
search = IntVar()


NAME = name.get()
```

```
    SUR = sur.get()
    EMAIL = email.get()
    BRANCH = branch.get()
    GANDER = munch.get()
    PYTHON = python.get()
    JAVA = java.get()
    C = c.get()
    D = d.get()
    CALCULATE = calculate.get()
    calculation2 = 4000
    flash=Label(groot,text="Registration Form", font=("arial",25),
bg="black",fg="white")
    flash.pack(side=TOP, fill=X)


    flash1 =Label(groot,text="Name:", font=("arial",17))
    flash1.place(x=300, y=150)


    flash2=Label(groot,text="Surname:", font=("arial",17))
    flash2.place(x=300, y=210)


    flash3=Label(groot,text="Email:", font=("arial",17))
    flash3.place(x=300, y=270)


    flash3=Label(groot,text="Gender:", font=("arial",17))
```

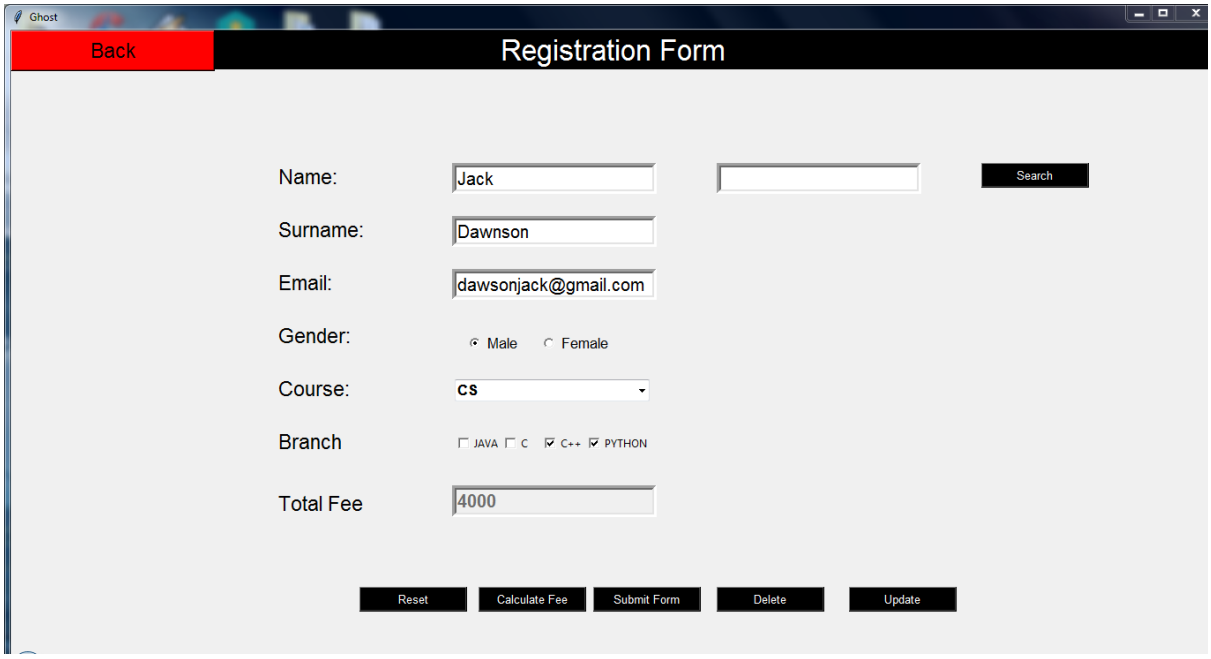
```
    flash3.place(x=300, y=330)


    flash4=Label(groot,text="Course:", font=("arial",17))
    flash4.place(x=300, y=390)


    flash4=Label(groot,text="Branch", font=("arial",17))
    flash4.place(x=300, y=450)


    flash4=Label(groot,text="Total Fee", font=("arial",17))
    flash4.place(x=300, y=520)


#==================================entryfield================
    lame5=Entry( groot,
textvar=calculate,state="readonly",width=20,font=("arial",15,"bold")
,bd=5)
    lame5.place(x=500, y=515)


    lame1=Entry(groot,bd=5, width=20,textvar=name ,font=("arial",15))
    lame1.place(x=500,y=150)


    lame2=Entry(groot,bd=5, width=20, textvar=sur ,font=("arial",15))
    lame2.place(x=500,y=210)


    lame3=Entry(groot,bd=5, width=20,textvar=email ,font=("arial",15))
    lame3.place(x=500,y=270)
```

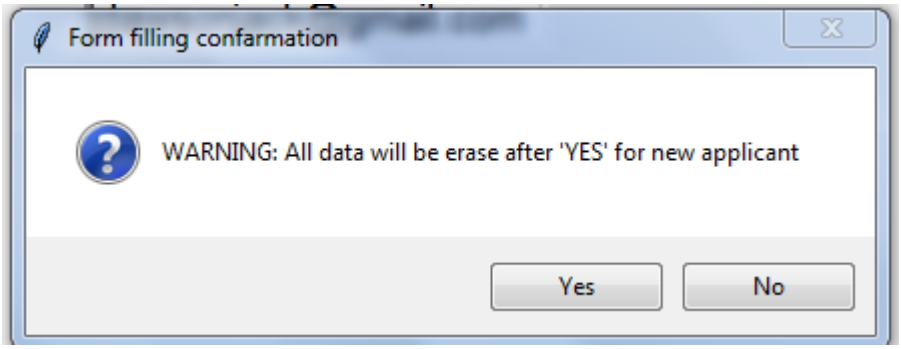
```
    lame4=Entry(groot,bd=5, text="enter roll no.",width=20,textvar=search
,font=("arial",15))

    lame4.place(x=800,y=150)

    search.set("")


#=========================radiobutton=====================

    radio1=Radiobutton(groot,text="Male", variable=munch, value=1
,font=("arial",13))

    radio1.place(x=515, y=340)


    radio2=Radiobutton(groot,text="Female", variable=munch, padx=10,
value=0 ,font=("arial",13))

    radio2.place(x=590, y=340)

    munch.set(3)


#==================================droplist=================

    box=ttk.Combobox(groot,textvariable=branch,state="readonly",
font=("arial",12,"bold"),width=22)

    box['values']=['SELECT','IP','CS','AI','IT']

    box.current(0)

    box.place(x=503,y=395)


#=========================checkbutton===============

    cb1=Checkbutton(groot,text="JAVA",variable=java)

    cb1.place(x=502,y=455 )
```

```
    cb1=Checkbutton(groot,text="C",variable=c)
    cb1.place(x=555,y=455 )

    cb1=Checkbutton(groot,text="C++",variable=d)
    cb1.place(x=600,y=455 ,)

    cb1=Checkbutton(groot,text="PYTHON",variable=python)
    cb1.place(x=650,y=455)
    python.set(0)
    java.set(0)
    c.set(0)
    d.set(0)
    def dis():
        groot.destroy()
        astral()

#======================button============================
    bttu1=Button(groot,text="Calculate
Fee",width=14,font=("arial",10),bg="black",fg="white"
,command=calculation)
    bttu1.place(x=530 , y=630)

    bttu12 = Button(groot, text="Back", width=17, font=("arial", 17),
bg="red",fg="black",command=dis )
    bttu12.place(x=0, y=0)
```

```
   bttu2=Button(groot,text="Submit
Form",width=14,font=("arial",10),bg="black",fg="white",command= msg  )
   bttu2.place(x=660 , y=630)

bttu3=Button(groot,text="Reset",width=14,font=("arial",10),bg="black",fg=
"white",command= golu )
   bttu3.place(x=395 , y=630)

bttu4=Button(groot,text="Search",width=14,font=("arial",10),bg="black",fg
="white" ,command=all )
   bttu4.place(x=1100 , y=150)

bttu4=Button(groot,text="Update",width=14,font=("arial",10),bg="black",f
g="white" ,command=update)
   bttu4.place(x=950 , y=630)

bttu5=Button(groot,text="Delete",width=14,font=("arial",10),bg="black",fg
="white",command=delete )
   bttu5.place(x=800 , y=630)

conn=sqlite3.connect("IP_Project.db")
with conn:
   cur=conn.cursor()

def ka():
```

```
    NAMEE=lame23.get()
    PHONE=lame24.get()
    PURPOSE=box2.get()
    conn=sqlite3.connect("IP_Project.db")
    with conn:
        cur=conn.cursor()
        cur.execute('INSERT INTO
ex(Name,Phone,Purpose,Date)VALUES(?,?,?,?)',(NAMEE,PHONE,PURPOSE,t
oday,))
        conn.commit()
def r():
    j()
    ka()
```

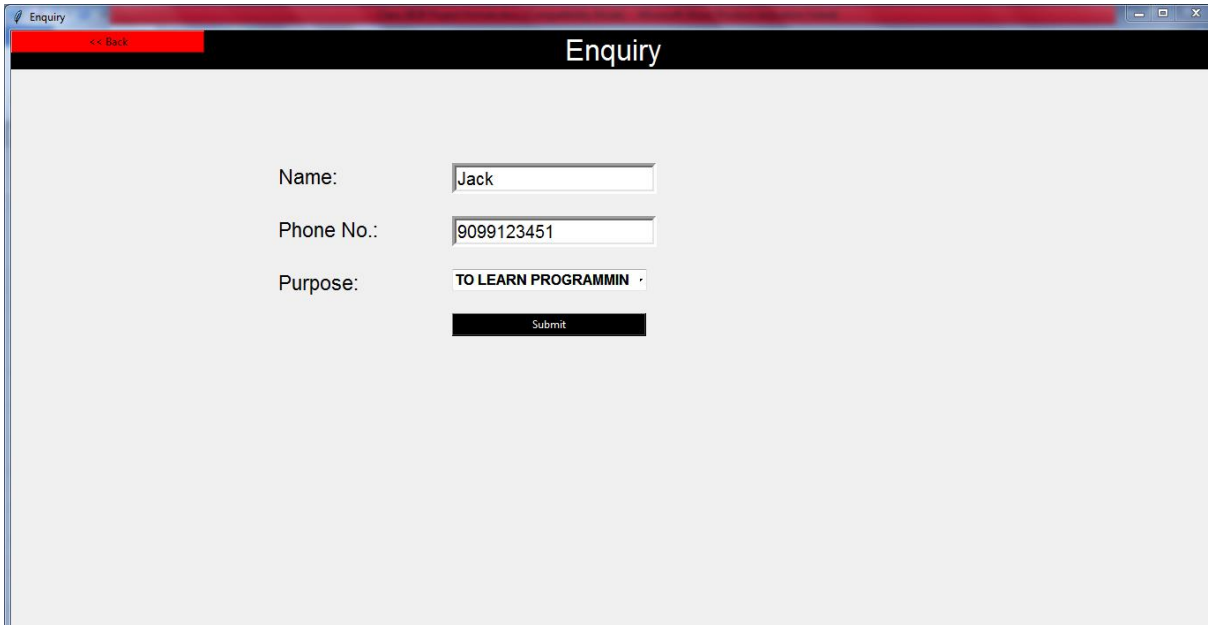




(f)Enquiry Window-

```
def eq1():
    eq1=Tk()
    eq1.title("Enquiry")
    eq1.geometry("1600x1000+0+0")
    purpose=StringVar()
    global lame23
    global lame24
    global box2
    def eq1destroy():
        eq1.destroy()
        astral()
    flash22 = Label(eq1, text="Enquiry", font=("arial", 25),
bg="black",fg="white")
    flash22.pack(side=TOP, fill=X)
    flash1 = Label(eq1, text="Name:", font=("arial", 17))
    flash1.place(x=300, y=150)
```

```
    flash2 = Label(eq1, text="Phone No.:", font=("arial", 17))
    flash2.place(x=300, y=210)


    flash3 = Label(eq1, text="Purpose:", font=("arial", 17))
    flash3.place(x=300, y=270)
    lame23 = Entry(eq1, bd=5, width=20, font=("arial", 15))
    lame23.place(x=500, y=150)
    bttu = Button(eq1, text="Submit", width=30, bg="black",fg="white",
command=r)
    bttu.place(x=500, y=320)
    bttu1=Button(eq1, text="<< Back", width=30,
bg="red",fg="black",command=eq1destroy)
    bttu1.place(x=0,y=0)
    lame24 = Entry(eq1, bd=5, width=20, font=("arial", 15))
    lame24.place(x=500, y=210)
    box2 = ttk.Combobox(eq1, textvariable=purpose, state="readonly",
font=("arial", 12, "bold"), width=22)
    box2['values'] = ['SELECT', 'TO LEARN PROGRAMMING', 'TO LEARN
MACHINE LEARNING', 'FEE DETAILS']
    box2.current(0)
    box2.place(x=500, y=270)
```



(g)Fee Details window-

```
def cat():
    J = IntVar()
    EZ = lame25.get()
    XT = lame26.get()
    YR = lame29.get()
    EZ=lame25.get()
    conn=sqlite3.connect("IP_Project.db")
    with conn:
        cur=conn.cursor()
        cur.execute('SELECT fee FROM dues WHERE id=?',(EZ,))
        for row24 in cur.fetchall():
          lame26.configure(state="normal")
          lame26.delete(0, END)
```

```
        lame26.insert(0,row24)
        lame26.configure(state="disable")
        cur.execute(' SELECT SUM(Installment) FROM machina WHERE id=?
GROUP BY id ', (EZ,))
        for row23 in cur.fetchall():
            lame27.delete(0, END)
            lame27.insert(0, row23)
            KL = lame27.get()
            J = int(float((lame26.get()))) - int(float((lame27.get())))
            lame28.configure(state="normal")
            lame28.delete(0, END)
            lame28.insert(0, z)
            print(row23)
            lame27.configure(state="disable")
            lame26.configure(state="disable")
            lame28.configure(state="disable")
            conn.commit()
            print(XT)
            print(EZ)
            print(today)

def reset2():
    lame26.configure(state="normal")
    lame25.configure(state="normal")
```

```
    lame27.configure(state="normal")
    lame28.configure(state="normal")
    lame29.configure(state="normal")
    lame26.delete(0,END )
    lame25.delete(0, END)
    lame27.delete(0,END)
    lame28.delete(0,END)
    lame29.delete(0,END)
    lame27.configure(state="disable")
    lame26.configure(state="disable")
    lame28.configure(state="disable")


def fee_add():
    J=IntVar()
    EZ=lame25.get()
    XT=lame26.get()
    YR=lame29.get()
    lame27.configure(state="normal")
    lame28.configure(state="normal")
    lame26.configure(state="normal")
    cur.execute('INSERT INTO machina(id , Total,Installment,
Date)VALUES(?,?,?,?)', (EZ, XT,YR, today,))
    cur.execute(' SELECT SUM(Installment) FROM machina WHERE id=?
GROUP BY id ',(EZ,))
    for row23 in cur.fetchall():
```

```
        lame27.delete(0,END)
        lame27.insert(0,row23)
        KL=lame27.get()
        J=int(float((lame26.get())))-int(float((lame27.get())))
        cur.execute('UPDATE machina SET Paid=? WHERE id=?' , (KL,EZ,))
        cur.execute('UPDATE machina SET REMAIN=? WHERE id=?',(J,EZ,))
        lame28.configure(state="normal")
        lame28.delete(0,END)
        lame28.insert(0,J)
        print(row23)
        lame27.configure(state="disable")
        lame26.configure(state="disable")
        lame28.configure(state="disable")
        conn.commit()
        print(XT)
        print(EZ)
        print(today)


def installment2():
    if int(lame29.index("end"))>int(0):
        fee_add()
    else:
        x=tkinter.messagebox.showinfo("No Fee Added","You Have Not Added
Any Fee ")
```

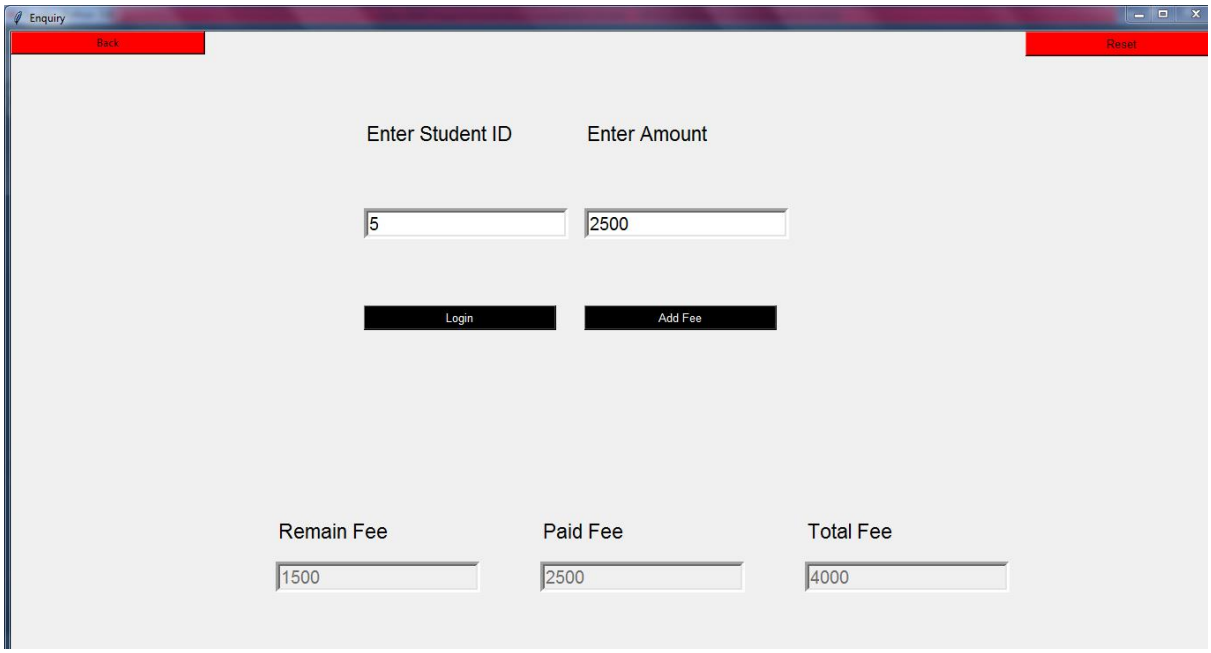
```
def j():
    Purpose=box2.get()
    print(Purpose)
def r():
    j()
    ka()


def window():
  global main
  global namee
  global phone
  global purpose
  global lame23
  global lame24
  global lame25
  global lame26
  global lame27
  global lame28
  global box2
  global key
  global fee3
  global KEY
  global ley
  global sey
```

```
  global ADDFEE
  global lame29


  main=Tk()
  main.geometry("1600x1000+0+0")
  main.title("Enquiry")
  namee=StringVar()
  phone=IntVar()
  purpose=StringVar()
  fe=StringVar()
  key=IntVar()
  ley=StringVar()
  sey=StringVar()
  def dis3():
     main.destroy()
     astral()


  bttu = Button(main, text="Back", width=30, bg="red",fg="black",
command=dis3)
  bttu.place(x=0, y=0)
  flash3=Label(main,text="Enter Student ID", font=("arial",17))
  flash3.place(x=400, y=100)
  flash3 = Label(main, text="Enter Amount", font=("arial", 17))
  flash3.place(x=650, y=100)
```

```
bttu22=Button(main,text="Login",width=26,font=("arial",10),bg="black",fg
="white",command=cat )

 bttu22.place(x=400, y=310)

 bttu23=Button(main,text="Add
Fee",width=26,font=("arial",10),bg="black",fg="white",command=installme
nt2 )

 bttu23.place(x=650 , y=310)

 lame29=Entry(main,bd=5, width=20 ,font=("arial",15))

 lame29.place(x=650,y=200)

 bttu28 = Button(main, text="Reset", width=26, font=("arial", 10),
bg="red",fg="black", command=reset2)

 bttu28.place(x=1150,y=0)


 flash31=Label(main,text="Total Fee", font=("arial",17))

 flash31.place(x=900, y=550)

 flash32=Label(main,text="Paid Fee", font=("arial",17))

 flash32.place(x=600, y=550)

 flash33=Label(main,text="Remain Fee", font=("arial",17))

 flash33.place(x=300, y=550)

 lame25=Entry(main,bd=5, width=20 ,font=("arial",15))

 lame25.place(x=400,y=200)

 lame26=Entry(main,bd=5, width=20 ,font=("arial",15))

 lame26.place(x=900,y=600)

 lame27=Entry(main,bd=5, width=20 ,font=("arial",15))

 lame27.place(x=600,y=600)
```

lame28=Entry(main,bd=5, width=20 ,font=("arial",15))

lame28.place(x=300,y=600)



(h)Functions-

#==========function==========

calculation2=4000

def calculation():

NAME = lame1.get()

SUR = lame2.get()

EMAIL = lame3.get()

BOX = box.get()

GANDER = munch.get()

PYTHON = python.get()

JAVA = java.get()

C = c.get()

```
D = d.get()
print(PYTHON)
print(GANDER)
CALCULATE = calculate.get()
if NAME==("") and  SUR==("")and  EMAIL==("") and BOX==("SELECT") and  GANDER==(3) and  JAVA==(0) and  PYTHON==(0) and C==(0) and D==(0):
        kal=tkinter.messagebox.showinfo(" Details Invalid","Fill All The Details")

else:
    global x
    if box.get()=="IP" and munch.get()==0:
        x=(calculation2-calculation2*20/100)
        lame5.configure(state="normal")
        lame5.delete(0,END)
        lame5.insert(0,x)
        lame5.configure(state="disable")
    if box.get()=="IP" and munch.get()==1:
        x=(calculation2-calculation2*10/100)
        lame5.configure(state="normal")
        lame5.delete(0, END)
        lame5.insert(0, x)
        lame5.configure(state="disable")
    if box.get()=="CS" and munch.get()==1:
```

```
        x=(calculation2)
        lame5.configure(state="normal")
        lame5.delete(0, END)
        lame5.insert(0, x)
        lame5.configure(state="disable")
    if box.get()=="CS" and munch.get()==0:
        x=(calculation2-calculation2*10/100)
        lame5.configure(state="normal")
        lame5.delete(0, END)
        lame5.insert(0, x)
        lame5.configure(state="disable")
    if box.get()=="IT" and munch.get()==0:
        x=(calculation2-calculation2*10/100)
        lame5.configure(state="normal")
        lame5.delete(0, END)
        lame5.insert(0, x)
        lame5.configure(state="disable")
    if box.get()=="IT" and munch.get()==1:
       x=(calculation2-calculation2*10/100)
       lame5.configure(state="normal")
       lame5.delete(0, END)
       lame5.insert(0, x)
       lame5.configure(state="disable")
    if box.get()=="AI" and munch.get()==1:
```

```
        x=(calculation2)
        lame5.configure(state="normal")
        lame5.delete(0, END)
        lame5.insert(0, x)
        lame5.configure(state="disable")
    if box.get()=="AI" and munch.get()==0:
        x=(calculation2-calculation2*10/100)
        lame5.configure(state="normal")
        lame5.delete(0, END)
        lame5.insert(0, x)
        lame5.configure(state="disable")


def msg():
 if branch.get()=="Select" or  munch.get()==3 or  ( python.get()==0 and
java.get==0 and c.get()==0 and d.get()==0):
     calculate.set("Please Fill All")
 if  "@" and ".com" not in lame3.get() :
    kal=tkinter.messagebox.showinfo(" Invalid Details","Enter Valid Email
Address")
    lame3.delete(0,END)
 else:
   msg=tkinter.messagebox.askyesno("Form filling confarmation","
WARNING: All data will be erase after 'YES' for new applicant" )
   if msg>0:
    NAME=lame1.get()
```

```
    SUR=lame2.get()
    EMAIL=lame3.get()
    BRANCH=box.get()
    GANDER=munch.get()
    PYTHON=python.get()
    JAVA=java.get()
    C=c.get()
    D=d.get()
    CALCULATE=calculate.get()
    conn=sqlite3.connect("IP_Project.db")
    with conn:
        cur=conn.cursor()
        cur.execute('INSERT INTO dues (name,sur, email, branch, munch,fee
,python,java,c,d )
VALUES(?,?,?,?,?,?,?,?,?,?)',(NAME,SUR,EMAIL,BRANCH,GANDER,CALCULAT
E,PYTHON,JAVA,C,D,))


        golu()


def golu():
    lame1.delete(0,END)
    lame2.delete(0,END)
    lame3.delete(0,END)
    box.set("Select")
    munch.set(3)
```

```
    python.set(0)
    java.set(0)
    c.set(0)
    d.set(0)
    calculate.set("")
    lame4.delete(0,END)

def search_id():
   SEARCH=lame4.get()
   conn=sqlite3.connect("IP_Project.db")
   with conn:
      cur=conn.cursor()
      cur.execute('SELECT name FROM dues WHERE id=?',(SEARCH))
      for row1 in cur.fetchone():
       name.set(row1)

def search_sur():
      SEARCH=lame4.get()
      conn=sqlite3.connect("IP_Project.db")
      with conn:
         cur=conn.cursor()
         cur.execute('SELECT sur FROM dues WHERE id=?',(SEARCH,))
         for row2 in cur.fetchone():
            sur.set(row2)
```

```
def search_email():
    SEARCH=lame4.get()
    conn=sqlite3.connect("IP_Project.db")
    with conn:
      cur=conn.cursor()
      cur.execute('SELECT email FROM dues WHERE id=?',(SEARCH,))
      for row3 in cur.fetchone():
        email.set(row3)

def search_branch():
    SEARCH=lame4.get()
    conn=sqlite3.connect("IP_Project.db")
    with conn:
      cur=conn.cursor()
      cur.execute('SELECT branch FROM dues WHERE id=?',(SEARCH,))
      for row4 in cur.fetchone():
        branch.set(row4)

def search_munch():
    SEARCH=lame4.get()
    conn=sqlite3.connect("IP_Project.db")
    with conn:
      cur=conn.cursor()
      cur.execute('SELECT munch FROM dues WHERE id=?',(SEARCH,))
```

```
        for row5 in cur.fetchone():
            munch.set(row5)


def search_course():
    SEARCH=lame4.get()
    conn=sqlite3.connect("IP_Project.db")
    with conn:
        cur=conn.cursor()
        cur.execute('SELECT python FROM dues WHERE id=?',(SEARCH,))
        for row6 in cur.fetchone():
            python.set(row6)
        cur.execute('SELECT java FROM dues WHERE id=?',(SEARCH,))
        for row7 in cur.fetchone():
            java.set(row7)
        cur.execute('SELECT c FROM dues WHERE id=?',(SEARCH,))
        for row8 in cur.fetchone():
            c.set(row8)
        cur.execute('SELECT d FROM dues WHERE id=?',(SEARCH,))
        for row9 in cur.fetchone():
            d.set(row9)
        cur.execute('SELECT fee FROM dues WHERE id=?',(SEARCH,))
        for row10 in cur.fetchone():
            calculate.set(row10)
```

```
def update():
   box1=tkinter.messagebox.askyesno("CONFIRMATION","if you update
you will be unable to see previous data again")
   if box1>0:
    SEARCH=lame4.get()
    NAME=lame1.get()
    SUR=lame2.get()
    EMAIL=lame3.get()
    BRANCH=box.get()
    GANDER=munch.get()
    PYTHON=python.get()
    JAVA=java.get()
    C=c.get()
    D=d.get()
    CALCULATE=lame5.get()


    conn=sqlite3.connect("IP_Project.db")
    with conn:
       cur=conn.cursor()
       cur.execute('UPDATE dues SET name=? WHERE
id=?',(NAME,SEARCH,))
       cur.execute('UPDATE dues SET sur=? WHERE id=?',(SUR,SEARCH,))
       cur.execute('UPDATE dues SET email=? WHERE
id=?',(EMAIL,SEARCH,))
```

```
        cur.execute('UPDATE dues SET branch=? WHERE
id=?',(BRANCH,SEARCH,))

        cur.execute('UPDATE dues SET munch=? WHERE
id=?',(GANDER,SEARCH,))

        cur.execute('UPDATE dues SET python=? WHERE
id=?',(PYTHON,SEARCH,))

        cur.execute('UPDATE dues SET java=? WHERE id=?',(JAVA,SEARCH,))

        cur.execute('UPDATE dues SET c=? WHERE id=?',(C,SEARCH,))

        cur.execute('UPDATE dues SET d=? WHERE id=?',(D,SEARCH,))

        conn.commit()


def delete():

   box=tkinter.messagebox.askyesno("WARNING","DATA WILL NOT BE
RECOVER AGAIN")

   if box>0:

      SEARCH = lame4.get()


      conn=sqlite3.connect("IP_Project.db")

      with conn:

          cur=conn.cursor()

          cur.execute("DELETE FROM dues WHERE id=?",(SEARCH))

          conn.commit()

          ex()


def all():
```

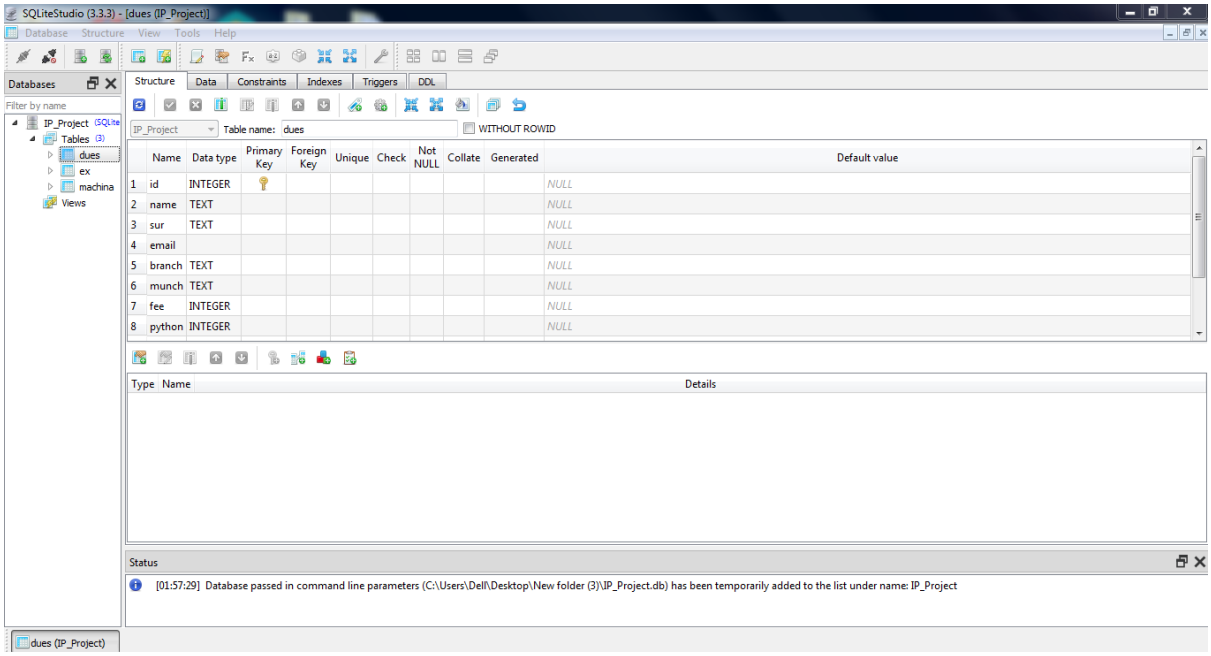
```
    search_id()
    search_sur()
    search_email()
    search_branch()
    search_munch()
    search_course()
    print("Student Details")
    print('='*50)
    print("Student ID:",lame4.get())
    print("Name:",lame1.get())
    print("Email:",lame3.get())
    print("Branch:",box.get())


INQUIRY=Button(text="Login",width=17,font=("arial",20),bg="black",fg="
white",command=lobby )
INQUIRY.place(x=560 , y=480)



firstw.mainloop()
```

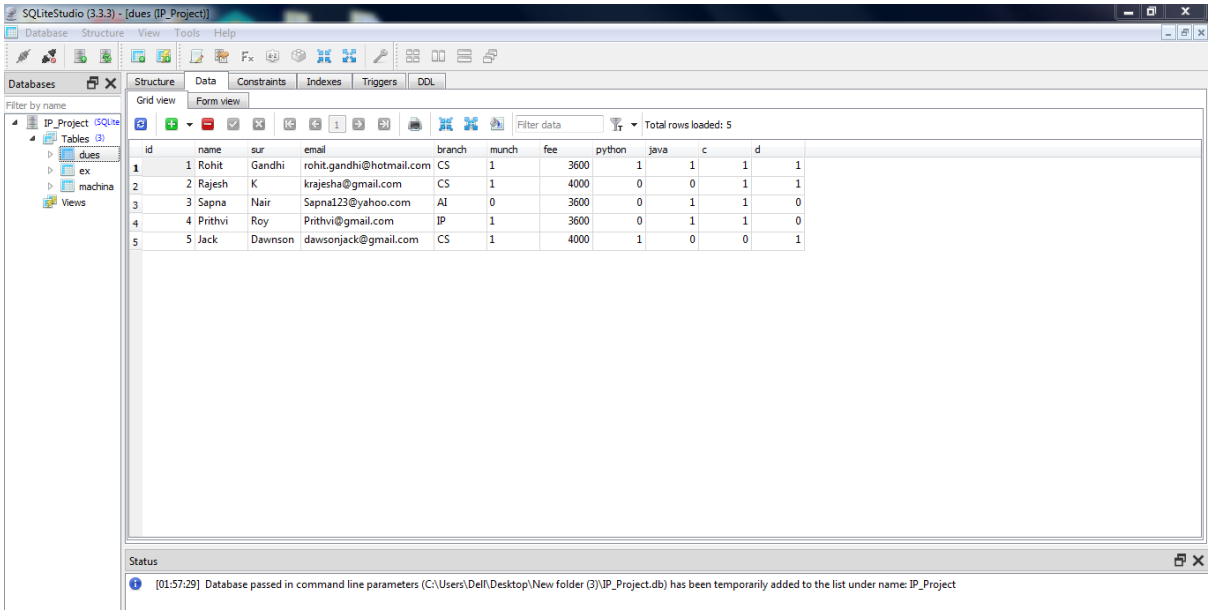
Software Coding-Back End

This Section explores the coding done for the back end .this code is supported by SQLite:

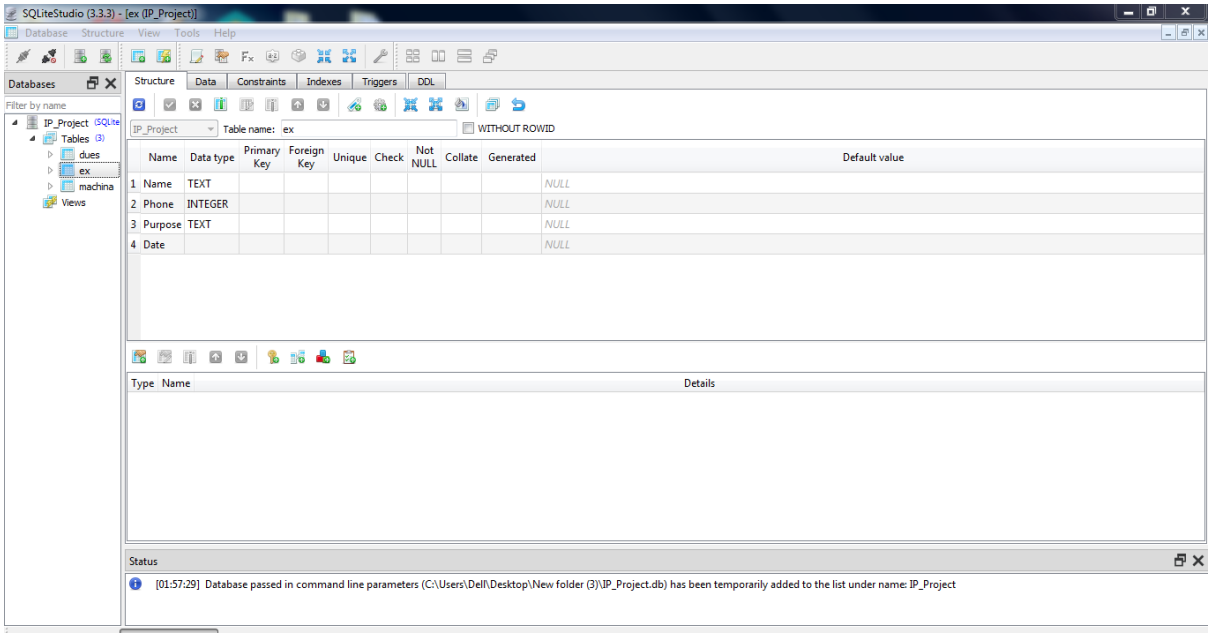
```
CREATE TABLE dues (
   id     INTEGER PRIMARY KEY AUTOINCREMENT,
```

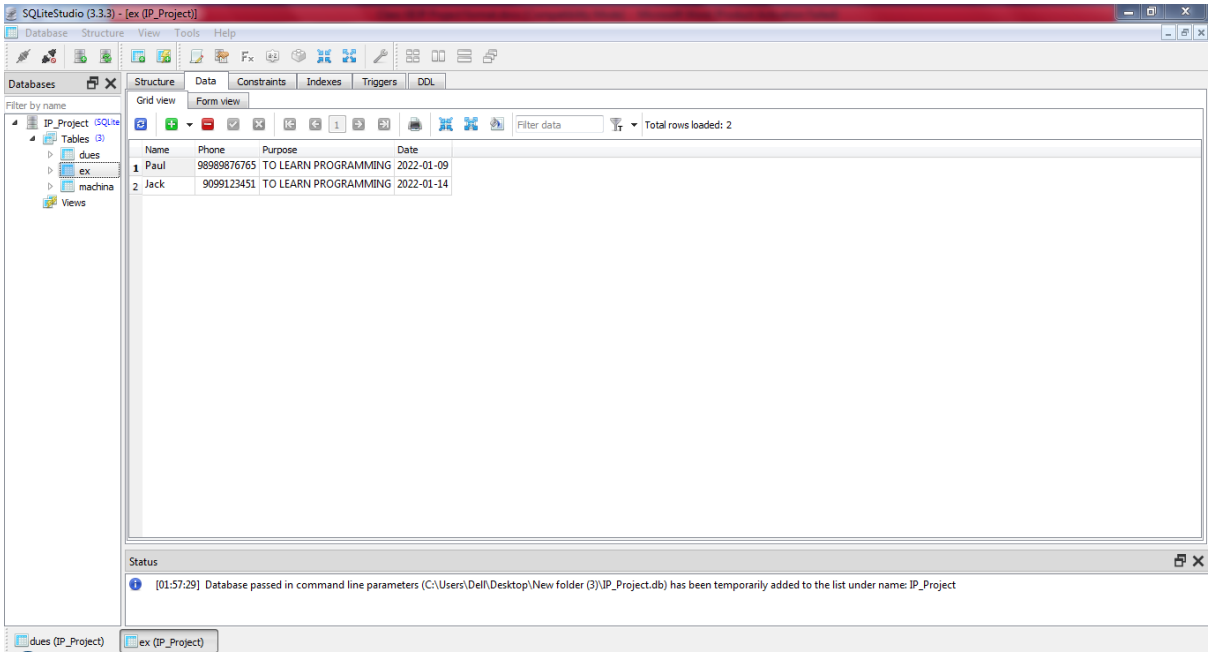
```
    name   TEXT,
    sur    TEXT,
    email,
    branch TEXT,
    munch  TEXT,
    fee    INTEGER,
    python INTEGER,
    java   INTEGER,
    c      INTEGER,
    d      INTEGER
);
```



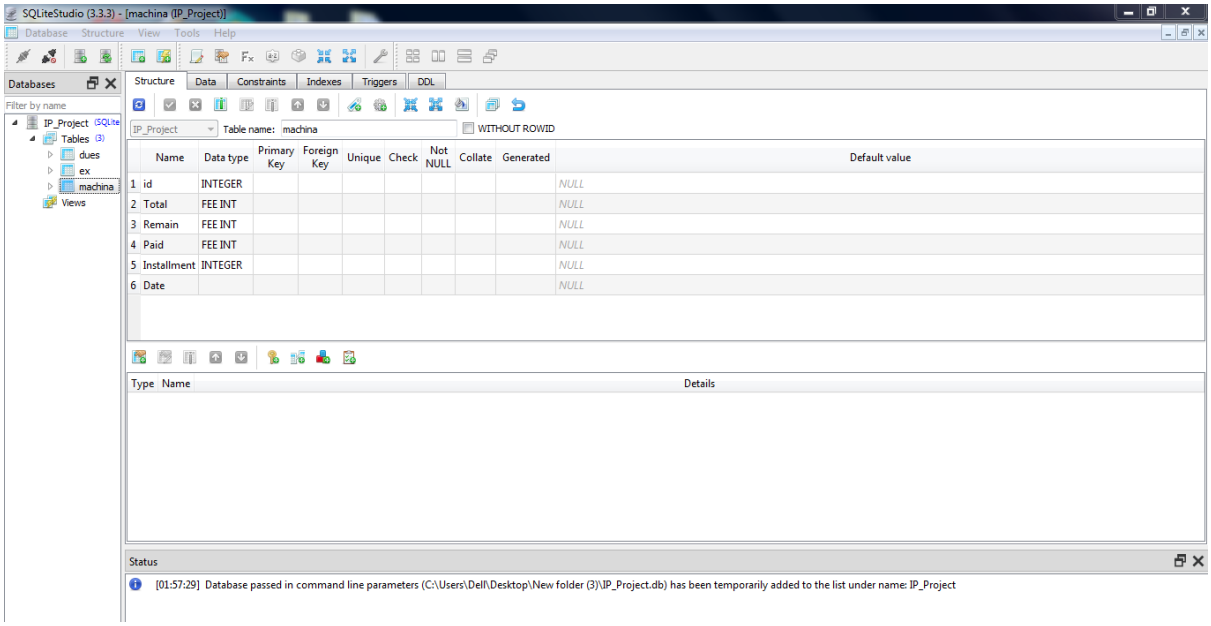


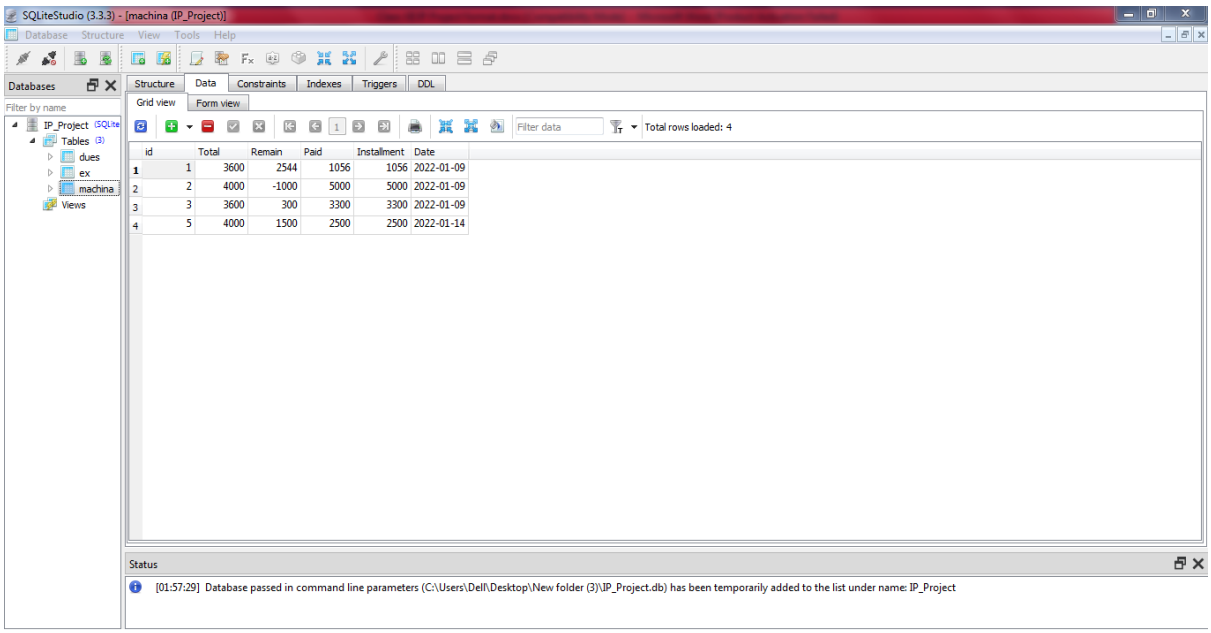

```
CREATE TABLE ex (
   Name    TEXT,
   Phone   INTEGER,
   Purpose TEXT,
   Date
);
```






```
CREATE TABLE machina (
   id         INTEGER,
   Total      [FEE INT],
   Remain     [FEE INT],
   Paid       [FEE INT],
   Installment INTEGER,
   Date
);
```





Known Limitations:

- The transactions are executed in off-line mode , hence on-line data for students, courses
  Capture and modification is not possible.
- Off-line reports of students and registrations cannot be generated due to batch mode execution.
- Excel export has not be developed for institutes, students due to some criticality.

Bibliography

This project was completed due to help from:

1. Informatics Practices-Sumita Arora
2. https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiEzqDA2a_1AhV-7XMBHQhfA8UQFnoECBQQAQ&url=https%3A%2F%2Fitsourcecode.com%2Ffree-projects%2Fpython-projects%2Fschool-management-system-project-in-python-with-source-code%2F&usg=AOvVaw2f40_wKlkeS2Rjy3fpF_IN
3. https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiEzqDA2a_1AhV-7XMBHQhfA8UQFnoECAUQAQ&url=https%3A%2F%2Fprojectsgeek.com%2F2014%2F02%2Fcomputer-institute-management-system-project.html&usg=AOvVaw2b0YmqSe8Vh0sxdal4mXP8