Name- Anand Verma

Email – vermanand007@gmail.com

Batch Name -12/07/2019 to 09/08/2019

Course – Python

Mobile No-7355982638

Python project Name-

1) *Music Player*
2) *Student Management System(Using pymysql)*
3) *Registration Form(Database browser)*

1) MUSIC PLAYER

 # CODE OF MUSIC PLAYER-

```
import os
import threading
import time
import tkinter.messagebox
from tkinter import *
from tkinter import filedialog

from mutagen.mp3 import MP3
from pygame import mixer

root = Tk()
root.geometry('880x278')
statusbar = Label(root, text="Welcome to ANAND Music
Player....(contact:7355982638,e-mail:vermanand007@gmail.com)",
relief=SUNKEN, anchor=W,bg='blue',fg='white')
statusbar.pack(side=BOTTOM, fill=X)
```

```
menubar = Menu(root)
root.config(menu=menubar,bg='pink')

subMenu = Menu(menubar, tearoff=0)

playlist = []

def browse_file():
    global filename_path
    filename_path = filedialog.askopenfilename()
    add_to_playlist(filename_path)


def add_to_playlist(filename):
    filename = os.path.basename(filename)
    index = 0
    playlistbox.insert(index, filename)
    playlist.insert(index, filename_path)
    index += 1


menubar.add_cascade(label="[File]", menu=subMenu)
subMenu.add_command(label="1. Open", command=browse_file)
subMenu.add_command(label="2. Exit", command=root.destroy)


def about_us():
    tkinter.messagebox.showinfo('About MUSIC', 'This is a music
player build using Python Tkinter by @attreyabhatt')


subMenu = Menu(menubar, tearoff=0)
menubar.add_cascade(label="[Help]", menu=subMenu)
subMenu.add_command(label="1. About Us", command=about_us)

mixer.init()

root.title("ANAND MUSIC PLAYER")
root.iconbitmap(r'music-player.ico')


leftframe = Frame(root,bg='pink')
leftframe.pack(side=LEFT, padx=40)

playlistbox =
Listbox(leftframe,height=13,width=70,bg='gray',fg='white')
playlistbox.pack()

addBtn = Button(leftframe, text="+ Add",
command=browse_file,bg='brown',fg='white',width=15,height=3)
addBtn.pack(side=LEFT)
```

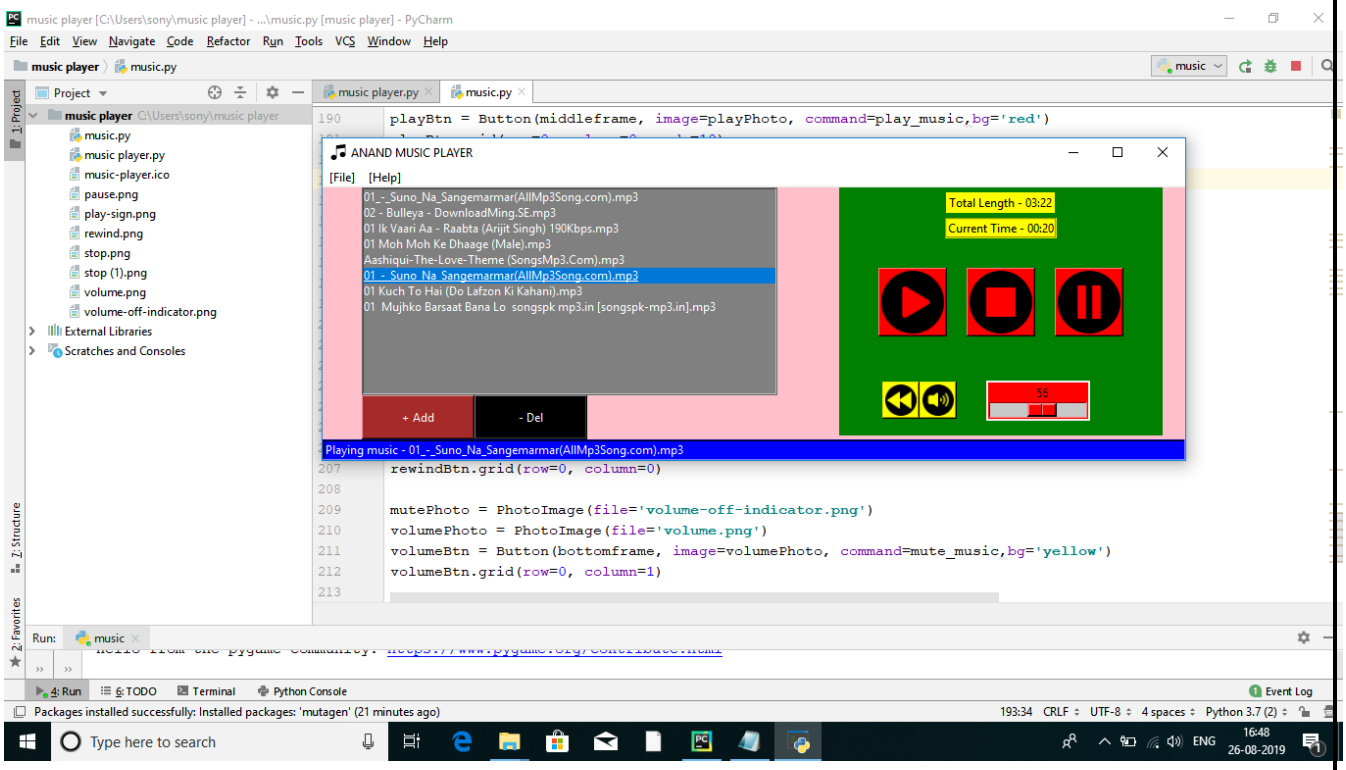
```
def del_song():
    selected_song = playlistbox.curselection()
    selected_song = int(selected_song[0])
    playlistbox.delete(selected_song)
    playlist.pop(selected_song)


delBtn = Button(leftframe, text="- Del",
command=del_song,bg='black',fg='white',width=15,height=3)
delBtn.pack(side=LEFT)

rightframe = Frame(root,bg='green')
rightframe.pack()

topframe = Frame(rightframe,bg='green')
topframe.pack()

lengthlabel = Label(topframe, text='Total Length : --:--
',bg='yellow')
lengthlabel.pack(pady=5)

currenttimelabel = Label(topframe, text='Current Time : --:--',
relief=GROOVE,bg='yellow')
currenttimelabel.pack()


def show_details(play_song):
    file_data = os.path.splitext(play_song)

    if file_data[1] == '.mp3':
        audio = MP3(play_song)
        total_length = audio.info.length
    else:
        a = mixer.Sound(play_song)
        total_length = a.get_length()
    mins, secs = divmod(total_length, 60)
    mins = round(mins)
    secs = round(secs)
    timeformat = '{:02d}:{:02d}'.format(mins, secs)
    lengthlabel['text'] = "Total Length" + ' - ' + timeformat

    t1 = threading.Thread(target=start_count,
args=(total_length,))
    t1.start()


def start_count(t):
    global paused
    current_time = 0
    while current_time <= t and mixer.music.get_busy():
        if paused:
            continue
        else:
```

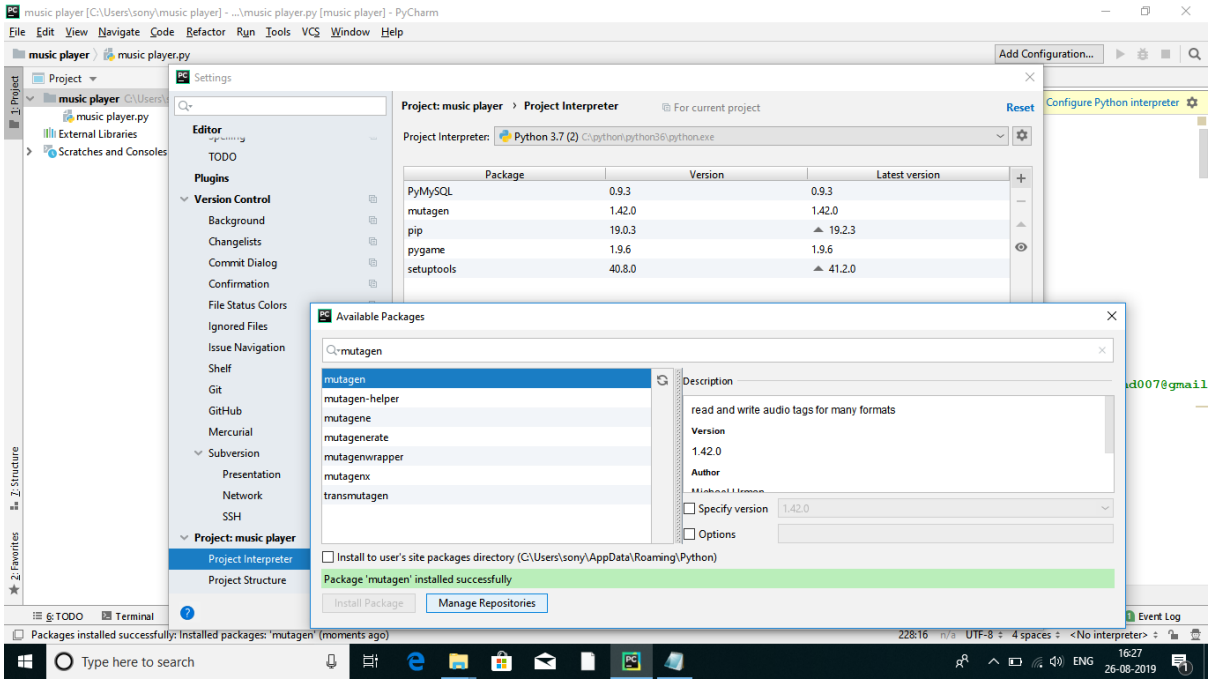
```
            mins, secs = divmod(current_time, 60)
            mins = round(mins)
            secs = round(secs)
            timeformat = '{:02d}:{:02d}'.format(mins, secs)
            currenttimelabel['text'] = "Current Time" + ' - ' +
timeformat
            time.sleep(1)
            current_time += 1


def play_music():
    global paused

    if paused:
        mixer.music.unpause()
        statusbar['text'] = "Music Resumed"
        paused = FALSE
    else:
        try:
            stop_music()
            time.sleep(1)
            selected_song = playlistbox.curselection()
            selected_song = int(selected_song[0])
            play_it = playlist[selected_song]
            mixer.music.load(play_it)
            mixer.music.play()
            statusbar['text'] = "Playing music" + ' - ' +
os.path.basename(play_it)
            show_details(play_it)
        except:
            tkinter.messagebox.showerror('File not found',
'Melody could not find the file. Please check again.')


def stop_music():
    mixer.music.stop()
    statusbar['text'] = "Music Stopped"


paused = FALSE


def pause_music():
    global paused
    paused = TRUE
    mixer.music.pause()
    statusbar['text'] = "Music Paused"


def rewind_music():
    play_music()
    statusbar['text'] = "Music Rewinded"
```

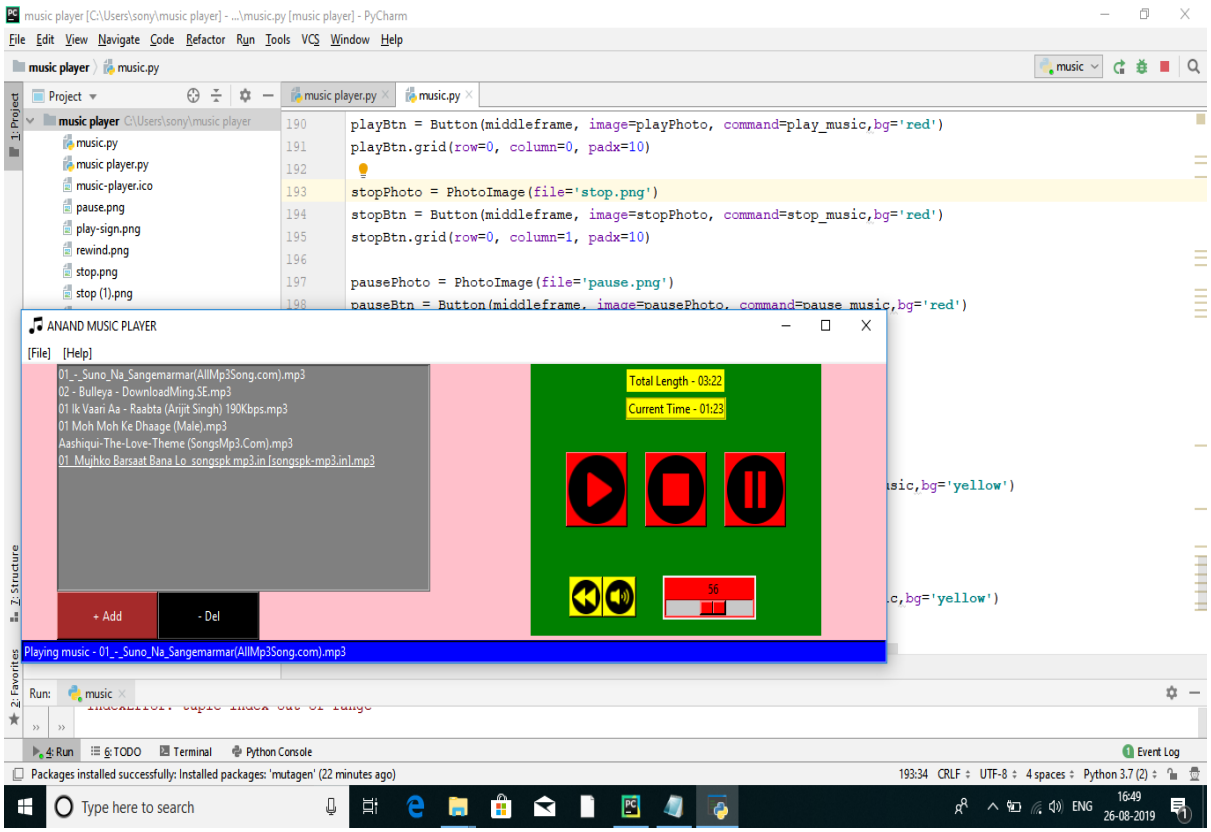
```
def set_vol(val):
    volume = int(val) / 100
    mixer.music.set_volume(volume)

muted = FALSE


def mute_music():
    global muted
    if muted:
        mixer.music.set_volume(0.7)
        volumeBtn.configure(image=volumePhoto)
        scale.set(70)
        muted = FALSE
    else:
        mixer.music.set_volume(0)
        volumeBtn.configure(image=mutePhoto)
        scale.set(0)
        muted = TRUE


middleframe = Frame(rightframe,bg='green')
middleframe.pack(pady=30, padx=30)

playPhoto = PhotoImage(file='play-sign.png')
playBtn = Button(middleframe, image=playPhoto,
command=play_music,bg='red')
playBtn.grid(row=0, column=0, padx=10)

stopPhoto = PhotoImage(file='stop.png')
stopBtn = Button(middleframe, image=stopPhoto,
command=stop_music,bg='red')
stopBtn.grid(row=0, column=1, padx=10)

pausePhoto = PhotoImage(file='pause.png')
pauseBtn = Button(middleframe, image=pausePhoto,
command=pause_music,bg='red')
pauseBtn.grid(row=0, column=2, padx=10)


bottomframe = Frame(rightframe,bg='green')
bottomframe.pack()

rewindPhoto = PhotoImage(file='rewind.png')
rewindBtn = Button(bottomframe, image=rewindPhoto,
command=rewind_music,bg='yellow')
rewindBtn.grid(row=0, column=0)

mutePhoto = PhotoImage(file='volume-off-indicator.png')
volumePhoto = PhotoImage(file='volume.png')
volumeBtn = Button(bottomframe, image=volumePhoto,
command=mute_music,bg='yellow')
```

```
volumeBtn.grid(row=0, column=1)

scale = Scale(bottomframe, from_=0, to=100, orient=HORIZONTAL,
command=set_vol,bg='red',width=14)
scale.set(70)
mixer.music.set_volume(0.7)
scale.grid(row=0, column=2, pady=15, padx=30)


def on_closing():
    stop_music()
    root.destroy()


root.protocol("WM_DELETE_WINDOW", on_closing)
root.mainloop()
```

SOME SCREENESHOT OF MUSIC PLAYER-







2) STUDENT MANGEMENT SYSTEM

CODE OF STUDENT MANAGEMENT SYSTEM-

```
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', 30, 'bold'), bg='yellow', fg='red')

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

        # ========================================all
variable==================================================

        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=590)


    m_title = Label(Manage_Frame, text="Manage Students", bg='crimson', font=('times new
roman', 30, 'bold'),

                fg='white')

    m_title.grid(row=0, columnspan=2, pady=20)


    lbl_roll = Label(Manage_Frame, text="Roll No.", bg='crimson', font=('times new roman', 20,
'bold'), fg='white')

    lbl_roll.grid(row=1, column=0, pady=10, padx=20, sticky='w')

    txt_roll = Entry(Manage_Frame, textvariable=self.Roll_No_var, font=('times new roman', 15,
'bold'), 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', font=('times new roman', 20,
'bold'), fg='white')

    lbl_name.grid(row=2, column=0, pady=10, padx=20, sticky='w')

    txt_name = Entry(Manage_Frame, textvariable=self.name_var, font=('times new roman', 15,
'bold'), 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', font=('times new roman', 20,
'bold'), fg='white')

    lbl_email.grid(row=3, column=0, pady=10, padx=20, sticky='w')

    txt_email = Entry(Manage_Frame, textvariable=self.email_var, font=('times new roman', 15,
'bold'), 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', font=('times new roman', 20,
'bold'), fg='white')

    lbl_gender.grid(row=4, column=0, pady=10, padx=20, sticky='w')

    combo_gender = ttk.Combobox(Manage_Frame, textvariable=self.gender_var, font=('times new
roman', 14, 'bold'),

                        state='readonly')

    combo_gender['values'] = ('male', 'female', 'other')

    combo_gender.grid(row=4, column=1, pady=10, padx=20, sticky='w')


    lbl_contact = Label(Manage_Frame, text="Contact", bg='crimson', font=('times new roman', 20,
'bold'),

                fg='white')

    lbl_contact.grid(row=5, column=0, pady=10, padx=20, sticky='w')

    txt_contact = Entry(Manage_Frame, textvariable=self.contact_var, font=('times new roman', 15,
'bold'), 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', font=('times new roman', 20,
'bold'), fg='white')

    lbl_dob.grid(row=6, column=0, pady=10, padx=20, sticky='w')

    txt_dob = Entry(Manage_Frame, textvariable=self.dob_var, font=('times new roman', 15, 'bold'),
bd=5,

            relief=GROOVE)

    txt_dob.grid(row=6, column=1, pady=10, padx=20, sticky='w')
```

```
    lbl_Address = Label(Manage_Frame, text="Address", bg='crimson', font=('times new roman', 20,
'bold'),

                        fg='white')

    lbl_Address.grid(row=7, column=0, pady=10, padx=20, sticky='w')


    self.txt_Address = Text(Manage_Frame, width=30, height=4, font=("", 10))

    self.txt_Address.grid(row=7, column=1, pady=10, padx=20, sticky='w')


    # ==================button frame======================

    btn_Frame = Frame(Manage_Frame, bd=4, relief=RIDGE, bg='crimson')

    btn_Frame.place(x=15, y=500, width=420)

    Addbtn = Button(btn_Frame, text='Add', width=10, command=self.add_students).grid(row=0,
column=0, padx=10,

                                                    pady=10)

    updatebtn = Button(btn_Frame, text='Update',
width=10,command=self.update_data).grid(row=0, padx=10, pady=10, column=1)

    deletebtn = Button(btn_Frame, text='Delete', width=10,command=self.delete_data).grid(row=0,
padx=10, pady=10, column=2)

    Clearbtn = Button(btn_Frame, text='Clear', width=10,command=self.clear).grid(row=0, padx=10,
pady=10, column=3)


    # ==========================detail
frame=============================================================

    Detail_Frame = Frame(self.root, bd=4, relief=RIDGE, bg='crimson')

    Detail_Frame.place(x=500, y=100, width=800, height=590)


    lbl_search = Label(Detail_Frame, text="Search By", bg='crimson', font=('times new roman', 20,
'bold'),

                       fg='white')

    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', 14, 'bold'), state='readonly')

    combo_search['values'] = ('Roll_No', 'Name', 'Contact')

    combo_search.grid(row=0, column=1, pady=10, padx=20, sticky='w')


    txt_search = Entry(Detail_Frame,textvariable=self.search_txt, width=20, font=('times new
roman', 10, 'bold'), bd=5, relief=GROOVE)

    txt_search.grid(row=0, column=2, pady=10, padx=20, sticky='w')


    search_btn = Button(Detail_Frame, text='Search', width=10,
pady=5,command=self.search_data).grid(row=0, column=3, padx=10, pady=10)

    search_btn = 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=======================================

    Table_Frame = Frame(Detail_Frame, bd=4, relief=RIDGE, bg='crimson')

    Table_Frame.place(x=10, y=70, width=770, height=500)


    scroll_y = Scrollbar(Table_Frame, orient=VERTICAL)

    scroll_x = Scrollbar(Table_Frame, orient=HORIZONTAL)

    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=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_students(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()=="" :
            messagebox.showerror("Error","All field are required!!!!")
        else:
            con = pymysql.connect(host='localhost', user='root', password='', database='stm')
            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_Address.get('1.0', END)
                                        ))
        con.commit()
        self.fetch_data()
        self.clear()
        con.close()
        messagebox.showinfo("Sucess","Record has been inserted")


def fetch_data(self):
        con = pymysql.connect(host='localhost', user='root', password='', database='stm')
        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 clear(self):
    self.Roll_No_var.set("")
    self.name_var.set("")
```

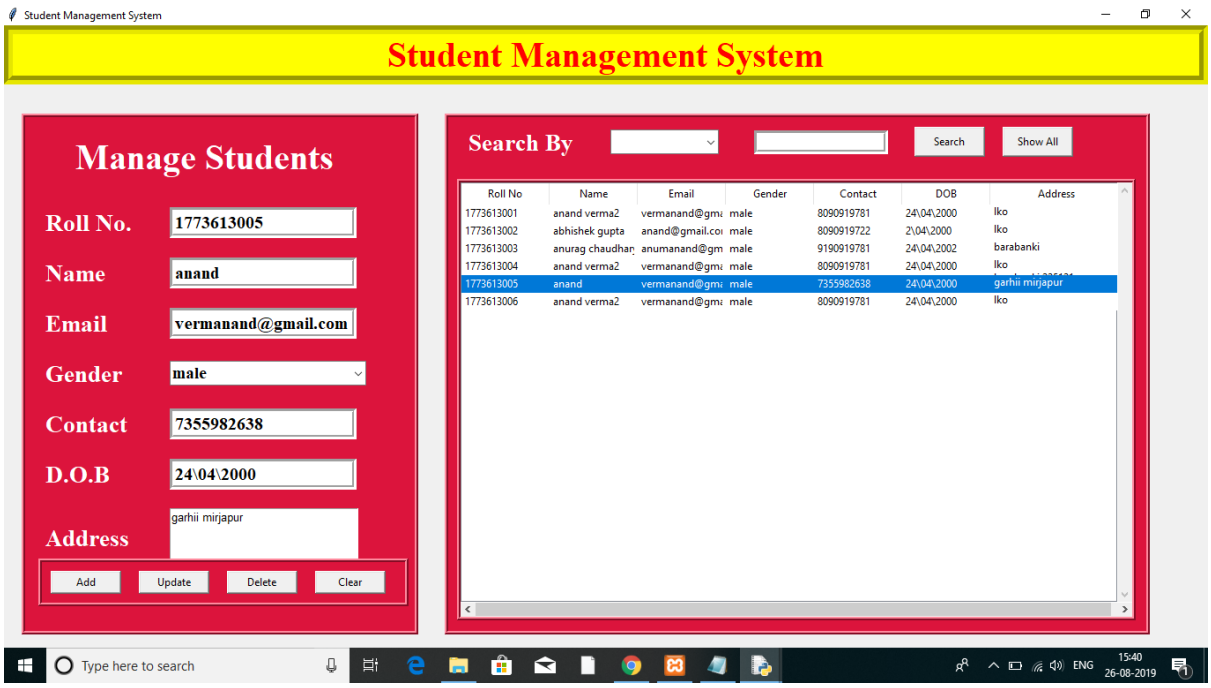
```
        self.email_var.set("")
        self.gender_var.set("")
        self.contact_var.set("")
        self.dob_var.set("")
        self.txt_Address.delete("1.0",END)

    def get_cursor(self,ev):
        curosor_row=self.Student_table.focus()
        contents=self.Student_table.item(curosor_row)
        row=contents['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):
        con = pymysql.connect(host='localhost', user='root', password='', database='stm')
        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(),
```

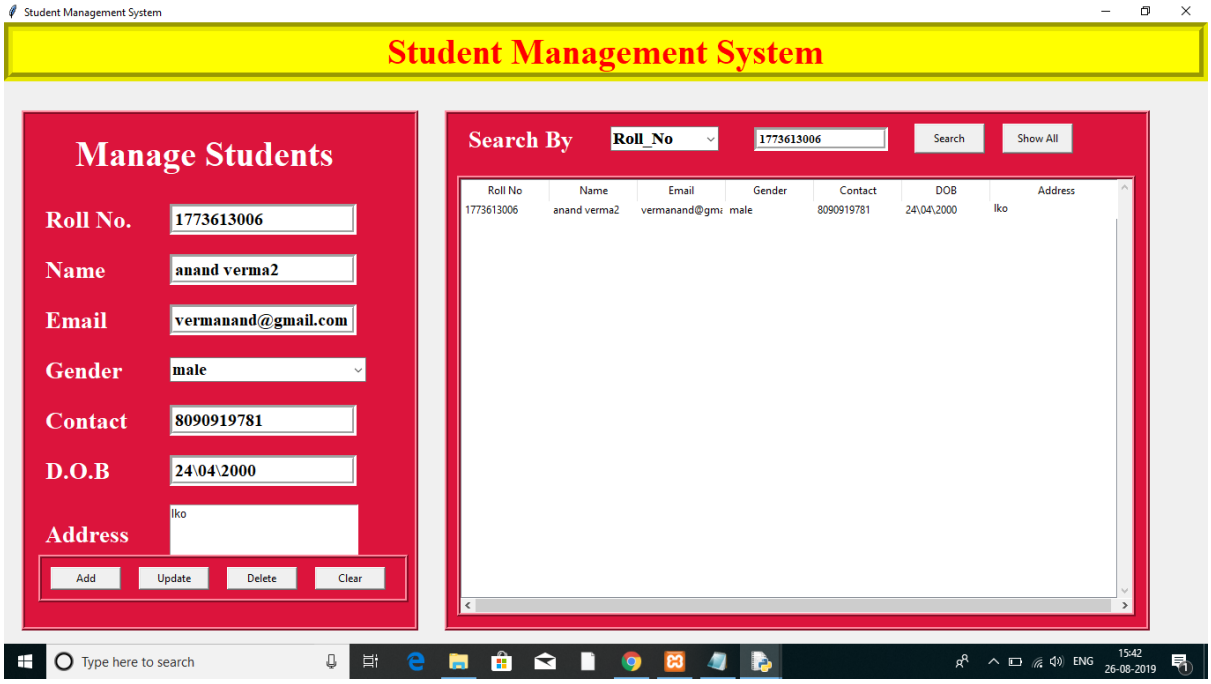
```
                                        self.dob_var.get(),
                                        self.txt_Address.get('1.0', END),
                                        self.Roll_No_var.get()
                                        ))
        con.commit()
        self.fetch_data()
        self.clear()
        con.close()


    def delete_data(self):
            con = pymysql.connect(host='localhost', user='root', password='', database='stm')
            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()
            con.close()


    def search_data(self):

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

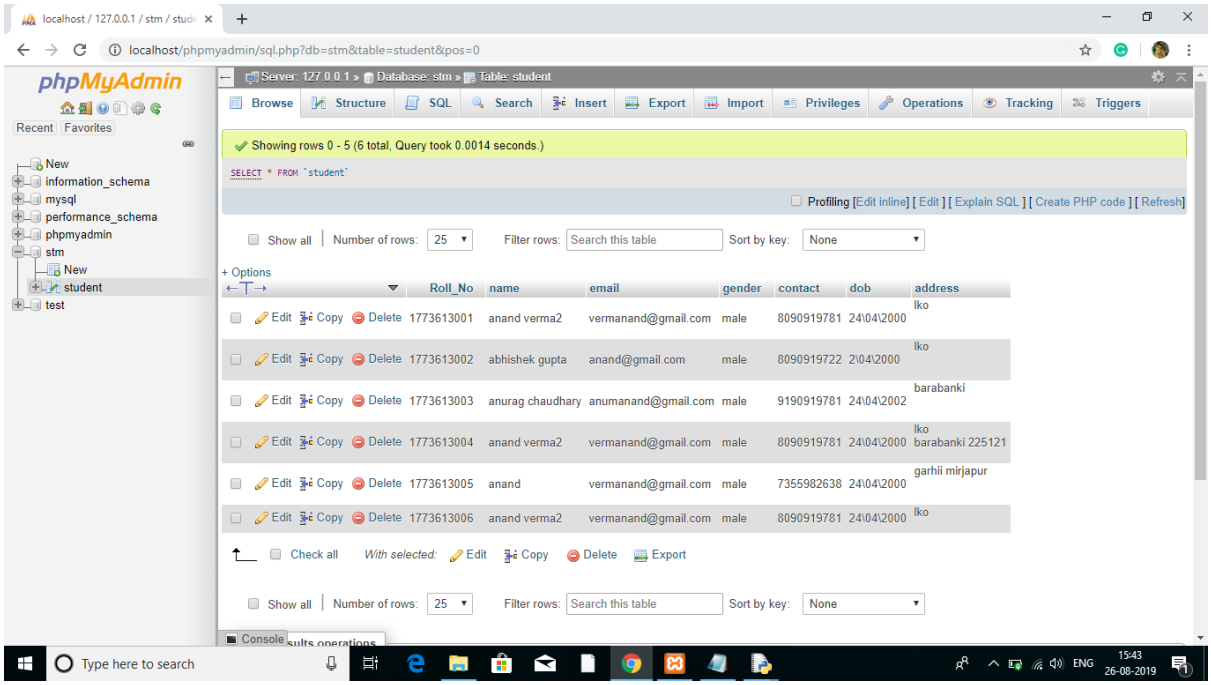
```
root = Tk()
ob = Student(root)
root.mainloop()
```

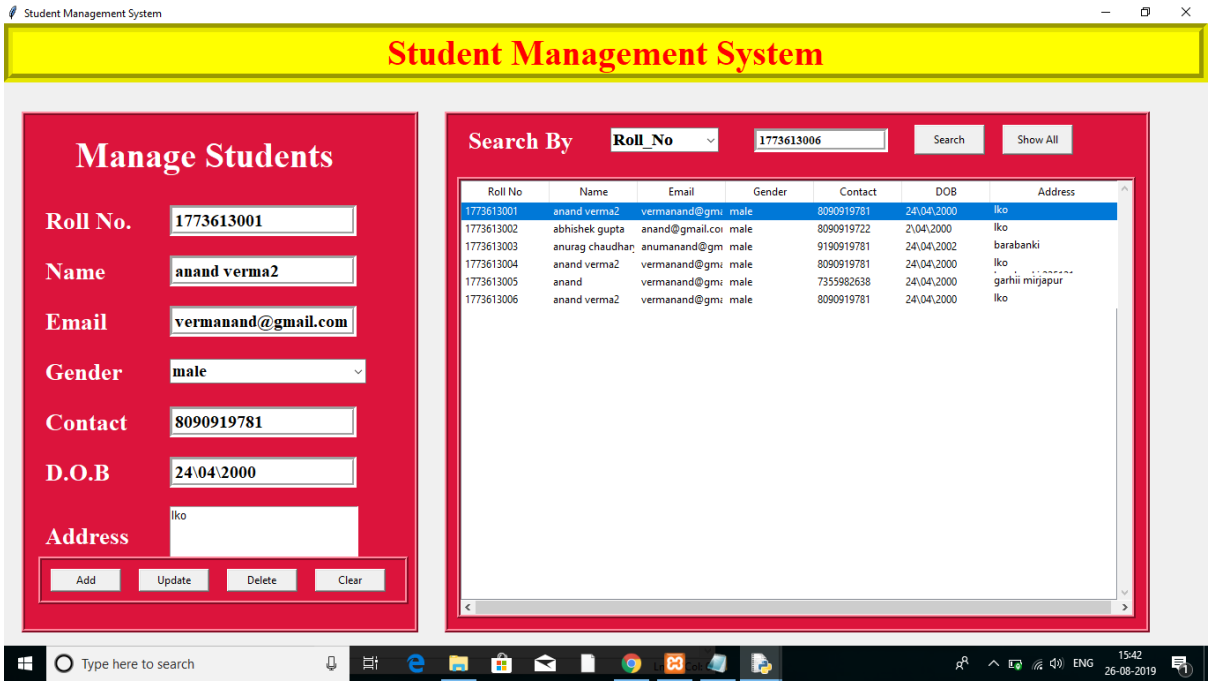
SOME SCREENSHOT OF STUDENT MANAGEMENT SYSTEM-




Student Management System
Student Management System
Manage Students
Roll No.
1773613006
Name
anand verma2
Email
vermanand@gmail.com
Gender
male
Contact
8090919781
D.O.B
24\04\2000
Address
Ikko
Add
Update
Delete
Clear
Search By
Roll_No
1773613006
Search
Show All
Roll No
Name
Email
Gender
Contact
DOB
Address
1773613006
anand verma2
vermanand@gm
male
8090919781
24\04\2000
Ikko
Type here to search
ENG
15:42
26-08-2019


XAMPP Control Panel v3.2.4 [Compiled: Jun 5th 2019]
XAMPP Control Panel v3.2.4
Modules
Service
Module
PID(s)
Port(s)
Actions
Apache
9648
7248
80, 443
MySQL
9540
3306
FileZilla
Mercury
Tomcat
Stop
Admin
Config
Logs
Start
Netstat
Shell
Explorer
Services
Help
Quit
15:28:10 [main] Initializing Control Panel
15:28:10 [main] Windows Version: Enterprise 64-bit
15:28:10 [main] XAMPP Version: 7.1.31
15:28:10 [main] Control Panel Version: 3.2.4 [Compiled: Jun 5th 2019]
15:28:10 [main] You are not running with administrator rights! This will work for
15:28:10 [main] most application stuff but whenever you do something with services
15:28:10 [main] there will be a security dialogue or things will break! So think
15:28:10 [main] about running this application with administrator rights!
15:28:10 [main] XAMPP Installation Directory: "c:\xampp\"
15:28:10 [main] Checking for prerequisites
15:28:26 [main] All prerequisites found
15:28:26 [main] Initializing Modules
15:28:26 [main] Starting Check-Timer
15:28:26 [main] Control Panel Ready
15:28:33 [Apache] Attempting to start Apache app...
15:28:34 [Apache] Status change detected: running
15:28:38 [mysql] Attempting to start MySQL app...
15:28:38 [mysql] Status change detected: running
Type here to search
ENG
15:43
26-08-2019


localhost / 127.0.0.1 / stm / student
localhost/phpmyadmin/sql.php?db=stm&table=student&pos=0
phpMyAdmin
Recent Favorites
New
information_schema
mysql
performance_schema
phpmyadmin
stm
New
student
test
Server: 127.0.0.1 » Database: stm » Table: student
Browse Structure SQL Search Insert Export Import Privileges Operations Tracking Triggers
Showing rows 0 - 5 (6 total, Query took 0.0014 seconds.)
SELECT * FROM `student`
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None
+ Options
Roll_No name email gender contact dob address
Edit Copy Delete 1773613001 anand verma2 vermanand@gmail.com male 8090919781 24\04\2000 lko
Edit Copy Delete 1773613002 abhishek gupta anand@gmail.com male 8090919722 2\04\2000 lko
Edit Copy Delete 1773613003 anurag chaudhary anumanand@gmail.com male 9190919781 24\04\2002 barabanki
Edit Copy Delete 1773613004 anand verma2 vermanand@gmail.com male 8090919781 24\04\2000 lko barabanki 225121
Edit Copy Delete 1773613005 anand vermanand@gmail.com male 7355982638 24\04\2000 garhii mirjapur
Edit Copy Delete 1773613006 anand verma2 vermanand@gmail.com male 8090919781 24\04\2000 lko
Check all With selected: Edit Copy Delete Export
Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None
Console
Type here to search
ENG 15:43 26-08-2019


Student Management System
Student Management System
Manage Students
Roll No. 1773613001
Name anand verma2
Email vermanand@gmail.com
Gender male
Contact 8090919781
D.O.B 24\04\2000
Address lko
Add Update Delete Clear
Search By Roll_No 1773613006 Search Show All
Roll No Name Email Gender Contact DOB Address
1773613001 anand verma2 vermanand@gm male 8090919781 24\04\2000 lko
1773613002 abhishek gupta anand@gmail.co male 8090919722 2\04\2000 lko
1773613003 anurag chaudhar anumanand@gm male 9190919781 24\04\2002 barabanki
1773613004 anand verma2 vermanand@gm male 8090919781 24\04\2000 lko
1773613005 anand vermanand@gm male 7355982638 24\04\2000 garhii mirjapur
1773613006 anand verma2 vermanand@gm male 8090919781 24\04\2000 lko
Type here to search
ENG 15:42 26-08-2019

3)REGISTRATION FORM

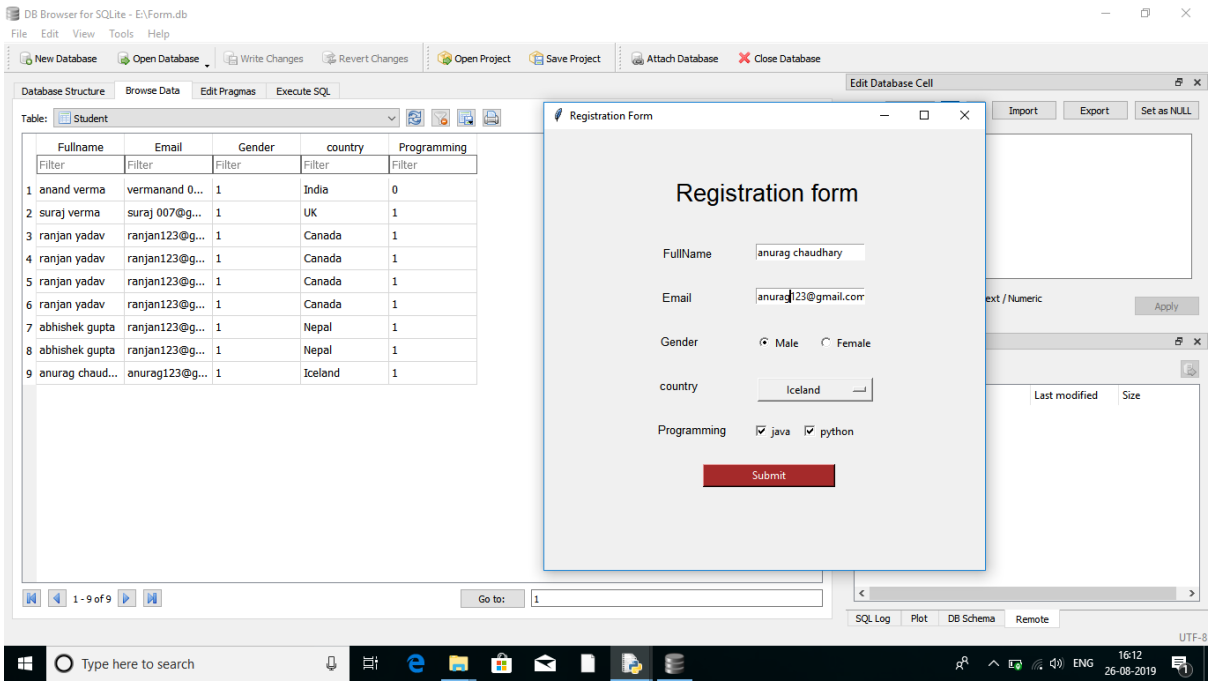
CODE OF REGISTRATION FORM-

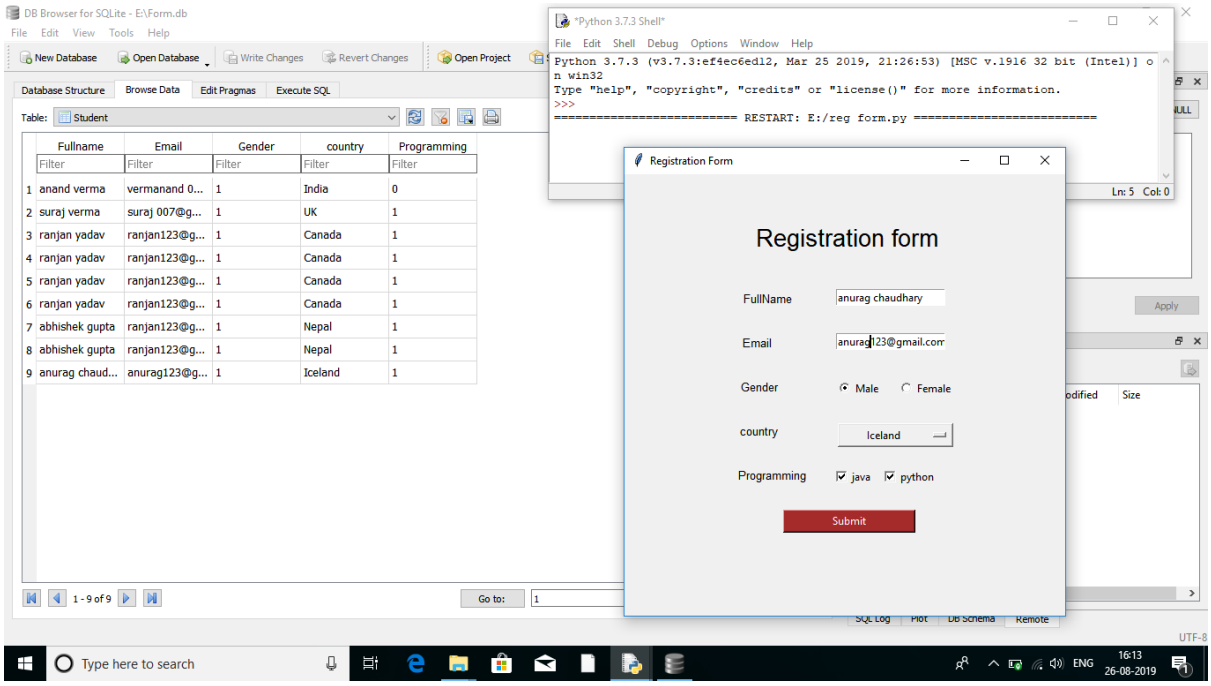
```
from tkinter import *
import sqlite3
root = Tk()
root.geometry('500x500')
root.title("Registration Form")
Fullname=StringVar()
Email=StringVar()
var = IntVar()
c=StringVar()
var1= IntVar()
def database():
    name1=Fullname.get()
    email=Email.get()
    gender=var.get()
    country=c.get()
    prog=var1.get()
    conn = sqlite3.connect('Form.db')
    with conn:
        cursor=conn.cursor()
    cursor.execute('CREATE TABLE IF NOT EXISTS Student (Fullname TEXT,Email TEXT,Gender
TEXT,country TEXT,Programming TEXT)')
    cursor.execute('INSERT INTO Student (FullName,Email,Gender,country,Programming)
VALUES(?,?,?,?,?)',(name1,email,gender,country,prog,))
```

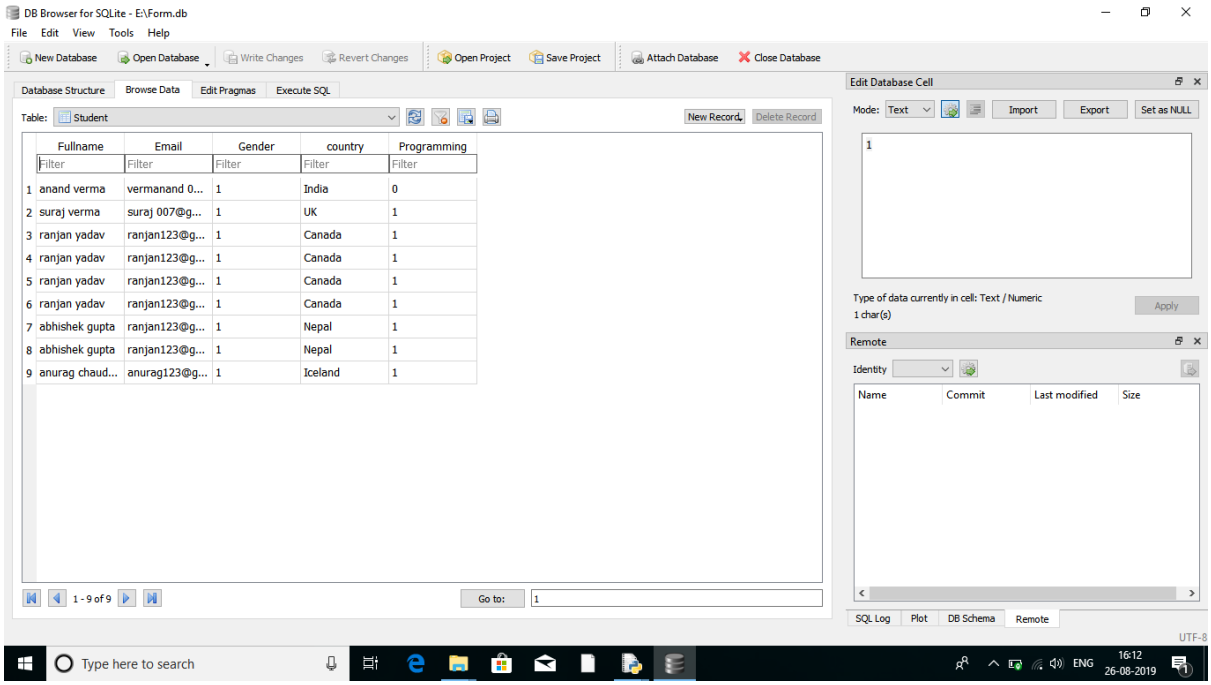
```
    conn.commit()
label_0 = Label(root, text="Registration form",width=20,font=("bold", 20))
label_0.place(x=90,y=53)
label_1 = Label(root, text="FullName",width=20,font=("bold", 10))
label_1.place(x=80,y=130)
entry_1 = Entry(root,textvar=Fullname)
entry_1.place(x=240,y=130)
label_2 = Label(root, text="Email",width=20,font=("bold", 10))
label_2.place(x=68,y=180)
entry_2 = Entry(root,textvar=Email)
entry_2.place(x=240,y=180)
label_3 = Label(root, text="Gender",width=20,font=("bold", 10))
label_3.place(x=70,y=230)
Radiobutton(root, text="Male",padx = 5, variable=var, value=1).place(x=235,y=230)
Radiobutton(root, text="Female",padx = 20, variable=var, value=2).place(x=290,y=230)
label_4 = Label(root, text="country",width=20,font=("bold", 10))
label_4.place(x=70,y=280)
list1 = ['Canada','India','UK','Nepal','Iceland','South Africa'];
droplist=OptionMenu(root,c, *list1)
droplist.config(width=15)
c.set('select your country')
droplist.place(x=240,y=280)
label_4 = Label(root, text="Programming",width=20,font=("bold", 10))
label_4.place(x=85,y=330)
var2= IntVar()
Checkbutton(root, text="java", variable=var1).place(x=235,y=330)
Checkbutton(root, text="python", variable=var2).place(x=290,y=330)
```

```
Button(root, text='Submit',width=20,bg='brown',fg='white',command=database).place(x=180,y=380)

root.mainloop()
```

#SOME SCREENSHOT OF REGISTRATION FORM-






DB Browser for SQLite - E:\Form.db
File Edit View Tools Help
New Database
Open Database
Write Changes
Revert Changes
Open Project
Save Project
Attach Database
Close Database
Database Structure
Browse Data
Edit Pragmas
Execute SQL
Table: Student
New Record
Delete Record
Fullname Email Gender country Programming
1 anand verma vermanand 0... 1 India 0
2 suraj verma suraj 007@g... 1 UK 1
3 ranjan yadav ranjan123@g... 1 Canada 1
4 ranjan yadav ranjan123@g... 1 Canada 1
5 ranjan yadav ranjan123@g... 1 Canada 1
6 ranjan yadav ranjan123@g... 1 Canada 1
7 abhishek gupta ranjan123@g... 1 Nepal 1
8 abhishek gupta ranjan123@g... 1 Nepal 1
9 anurag chaud... anurag123@g... 1 Iceland 1
1 - 9 of 9
Go to: 1
Edit Database Cell
Mode: Text
Import
Export
Set as NULL
Type of data currently in cell: Text / Numeric
1 char(s)
Apply
Remote
Identity
Name Commit Last modified Size
SQL Log
Plot
DB Schema
Remote
UTF-8
Type here to search
ENG
16:12
26-08-2019