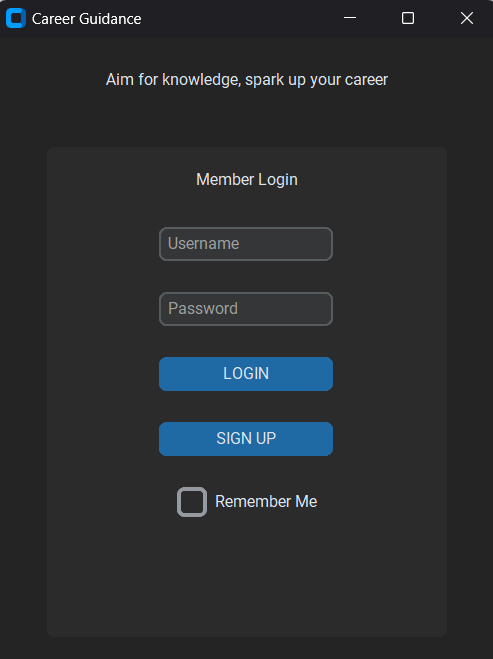
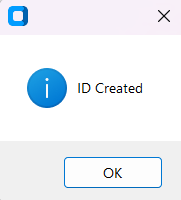
***LOGIN FORM FOR A MEMBER USING DATABASE (sqlite3):***

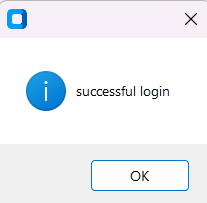
After running the code successfully, we can able to see the above LOGIN PAGE to be opened.

STEP 1: For Sign up give the correct credentials and then click sign up.

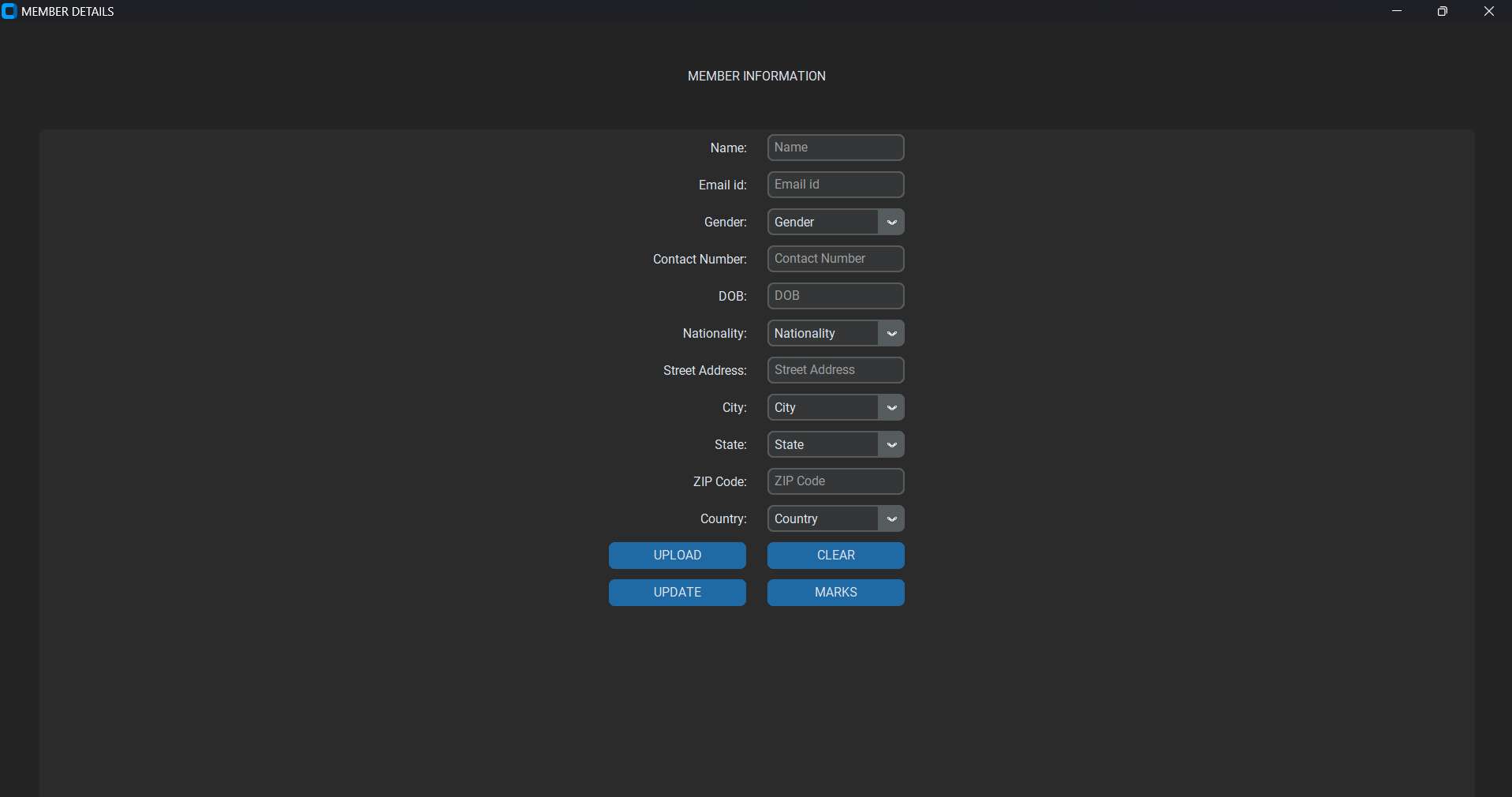
Then you can receive the below dialog box

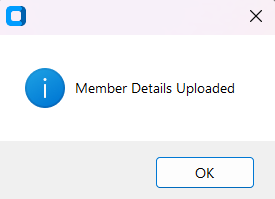


STEP 2: Sign in using the same credentials to login the page.



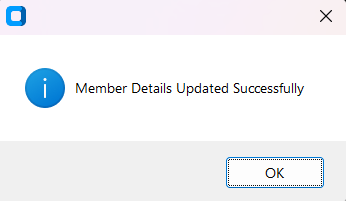
STEP 3: After a successful login. A member details page will be opened. Fill the details and click UPLOAD.



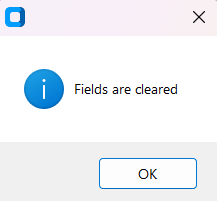


Above image can be displayed after the member details are uploaded in to the database successfully.

**UPDATE BUTTON**: Used to update the Existing data with the current one from the database.



**CLEAR BUTTON**: Used to clear all the fields and able to fill the data.



STEP 4: Click on the MARKS BUTTON to open a new page to fill the mark details as below.

**UPDATE BUTTON**: Used to update the Existing data with the current one from the database.

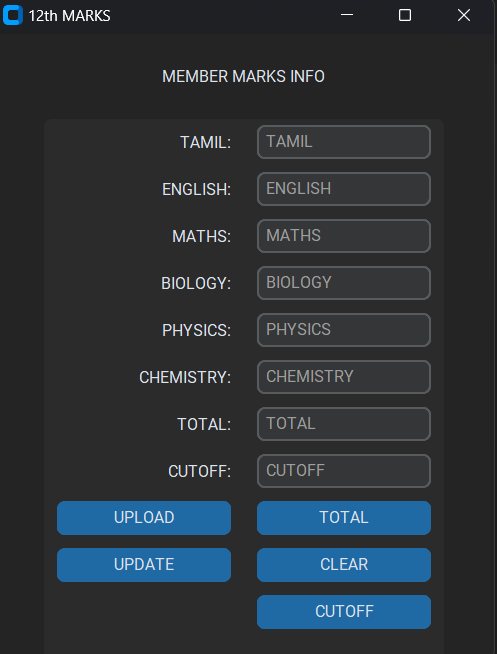
**UPLOAD BUTTON**: Used to Upload the data into the database.

**CLEAR BUTTON**: Used to clear all the fields and able to fill the data.

**TOTAL BUTTON**: Used to total all the subject and give us a percentage.

**CUTOFF BUTTON**: Gives the cutoff mark of the member based upon the marks

respectively.



CODE :

import customtkinter as ctk

import tkinter.messagebox as tkmb

import sqlite3

from tkinter import StringVar

from tkinter import END

from PIL import Image,ImageTk

ctk.set\_appearance\_mode("dark")

ctk.set\_default\_color\_theme("blue")

class App(ctk.CTk):

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

super().\_\_init\_\_()

self.geometry("400x500")

self.title("Career Guidance")

self.connector = sqlite3.connect('career12.db')

self.cursorObj = self.connector.cursor()

self.cursorObj.execute("CREATE TABLE if not exists CREDENTIALS(username TEXT PRIMARY KEY,password TEXT)")

self.label = ctk.CTkLabel(self, text="Aim for knowledge, spark up your career")

self.label.pack(pady=20)

self.frame = ctk.CTkFrame(self)

self.frame.pack(pady=20, padx=40, fill='both', expand=True)

self.label\_login = ctk.CTkLabel(master=self.frame, text='Member Login')

self.label\_login.pack(pady=12, padx=10)

self.E1 = ctk.CTkEntry(master=self.frame, placeholder\_text="Username")

self.E1.pack(pady=12, padx=10)

self.E2 = ctk.CTkEntry(master=self.frame, placeholder\_text="Password", show="\*")

self.E2.pack(pady=12, padx=10)

self.button\_login = ctk.CTkButton(master=self.frame, text='LOGIN', command=self.Login)

self.button\_login.pack(pady=12, padx=10)

self.button\_signup = ctk.CTkButton(master=self.frame, text='SIGN UP', command=self.Sign\_Up)

self.button\_signup.pack(pady=12, padx=10)

self.checkbox = ctk.CTkCheckBox(master=self.frame, text='Remember Me')

self.checkbox.pack(pady=12, padx=10)

def Sign\_Up(self):

s = sqlite3.connect('career12.db')

print("Connected")

f = self.E1.get()

g = self.E2.get()

if (f == "") or (g == ""):

tkmb.showinfo("", "Fill the Empty Field")

else:

s.execute("INSERT INTO CREDENTIALS(username,password) VALUES(?,?)", (f, g))

s.commit()

tkmb.showinfo("", "ID Created")

print("ID Created")

def Login(self):

c = sqlite3.connect('career12.db')

print("Connected")

s = c.execute("select \* from CREDENTIALS")

d = self.E1.get()

e = self.E2.get()

if (d == "") or (e == ""):

tkmb.showinfo("", "Fill the Empty Field")

return

for i in s:

a = i[0]

b = i[1]

if (a == d) and (b == e):

tkmb.showinfo("", "successful login")

self.show\_member\_details()

break

else:

tkmb.showinfo("", "Enter Correct UserName and Password")

def show\_member\_details(self):

app1 = ctk.CTk()

app1.geometry("400x500")

app1.title("MEMBER DETAILS")

label = ctk.CTkLabel(app1, text="MEMBER INFORMATION")

label.pack(padx=20,pady=40)

frame = ctk.CTkFrame(master=app1)

frame.pack(padx=40, fill='both', expand=True)

label\_name = ctk.CTkLabel(master=frame, text="Name:")

label\_name.grid(row=0, column=0, padx=10, pady=5, sticky="e")

self.E7 = ctk.CTkEntry(master=frame, placeholder\_text="Name")

self.E7.grid(row=0, column=1, padx=10, pady=5,sticky="w")

label\_contact = ctk.CTkLabel(master=frame, text="Contact Number:")

label\_contact.grid(row=3, column=0, padx=10, pady=5, sticky="e")

self.E8 = ctk.CTkEntry(master=frame, placeholder\_text="Contact Number")

self.E8.grid(row=3, column=1, padx=10, pady=5,sticky="w")

label\_gender = ctk.CTkLabel(master=frame, text="Gender:")

label\_gender.grid(row=2, column=0, padx=10, pady=5, sticky="e")

Gender = ["Male", "Female", "Transgender", "Prefer not to say"]

self.E9 = ctk.CTkComboBox(master=frame, values=Gender)

self.E9.set("Gender")

self.E9.grid(row=2, column=1, padx=10, pady=5,sticky="w")

label\_email = ctk.CTkLabel(master=frame, text="Email id:")

label\_email.grid(row=1, column=0, padx=10, pady=5, sticky="e")

self.E10 = ctk.CTkEntry(master=frame, placeholder\_text="Email id")

self.E10.grid(row=1, column=1, padx=10, pady=5,sticky="w")

label\_dob = ctk.CTkLabel(master=frame, text="DOB:")

label\_dob.grid(row=4, column=0, padx=10, pady=5, sticky="e")

self.E11 = ctk.CTkEntry(master=frame, placeholder\_text="DOB")

self.E11.grid(row=4, column=1, padx=10, pady=5,sticky="w")

label\_nationality = ctk.CTkLabel(master=frame, text="Nationality:")

label\_nationality.grid(row=5, column=0, padx=10, pady=5, sticky="e")

Nationality = ["Indian", "Others"]

self.E12 = ctk.CTkComboBox(master=frame, values=Nationality)

self.E12.set("Nationality")

self.E12.grid(row=5, column=1, padx=10, pady=5,sticky="w")

label\_address = ctk.CTkLabel(master=frame, text="Street Address:")

label\_address.grid(row=6, column=0, padx=10, pady=5, sticky="e")

self.E13 = ctk.CTkEntry(master=frame, placeholder\_text="Street Address")

self.E13.grid(row=6, column=1, padx=10, pady=5,sticky="w")

label\_city = ctk.CTkLabel(master=frame, text="City:")

label\_city.grid(row=7, column=0, padx=10, pady=5, sticky="e")

City = ["Chennai", "Coimbatore", "Madurai", "Tiruchirappalli", "Salem", "Tiruppur", "Erode", "Vellore", "Thoothukudi", "Dindigul", "Thanjavur", "Tirunelveli", "Kanchipuram", "Ooty", "Cuddalore", "Kumbakonam", "Karur", "Hosur", "Nagercoil", "Theni", "Sivakasi", "Neyveli", "Rajapalayam", "Pudukottai", "Namakkal", "Nagapattinam", "Tiruvannamalai", "Pollachi", "Ambur", "Krishnagiri", "Ranipet", "Arakkonam", "Viluppuram", "Tindivanam", "Perambalur", "Virudhunagar", "Arani", "Dharmapuri", "Palani", "Chidambaram", "Kovilpatti", "Karaikudi", "Mannargudi", "Mayiladuthurai", "Tiruchengode", "Thiruvallur", "Mettur", "Nellikuppam", "Sivaganga", "Gudiyatham"]

self.E14 = ctk.CTkComboBox(master=frame, values=City)

self.E14.set("City")

self.E14.grid(row=7, column=1, padx=10, pady=5,sticky="w")

label\_state = ctk.CTkLabel(master=frame, text="State:")

label\_state.grid(row=8, column=0, padx=10, pady=5, sticky="e")

State = ["Andhra Pradesh", "Arunachal Pradesh", "Assam", "Bihar", "Chhattisgarh", "Goa", "Gujarat", "Haryana", "Himachal Pradesh", "Jharkhand", "Karnataka", "Kerala", "Madhya Pradesh", "Maharashtra", "Manipur", "Meghalaya", "Mizoram", "Nagaland", "Odisha", "Punjab", "Rajasthan", "Sikkim", "Tamil Nadu", "Telangana", "Tripura", "Uttar Pradesh", "Uttarakhand", "West Bengal", "Andaman and Nicobar Islands", "Chandigarh", "Dadra and Nagar Haveli and Daman and Diu", "Delhi", "Lakshadweep", "Puducherry"]

self.E15 = ctk.CTkComboBox(master=frame, values=State)

self.E15.set("State")

self.E15.grid(row=8, column=1, padx=10, pady=5,sticky="w")

label\_zip = ctk.CTkLabel(master=frame, text="ZIP Code:")

label\_zip.grid(row=9, column=0, padx=10, pady=5, sticky="e")

self.E16 = ctk.CTkEntry(master=frame, placeholder\_text="ZIP Code")

self.E16.grid(row=9, column=1, padx=10, pady=5,sticky="w")

label\_country = ctk.CTkLabel(master=frame, text="Country:")

label\_country.grid(row=10, column=0, padx=10, pady=5, sticky="e")

Country = ["Afghanistan", "Albania", "Algeria", "Andorra", "Angola", "Antigua and Barbuda", "Argentina", "Armenia", "Australia", "Austria", "Azerbaijan", "Bahamas", "Bahrain", "Bangladesh", "Barbados", "Belarus", "Belgium", "Belize", "Benin", "Bhutan", "Bolivia", "Bosnia and Herzegovina", "Botswana", "Brazil", "Brunei", "Bulgaria", "Burkina Faso", "Burundi", "Cabo Verde", "Cambodia", "Cameroon", "Canada", "Central African Republic", "Chad", "Chile", "China", "Colombia", "Comoros", "Congo", "Costa Rica", "Croatia", "Cuba", "Cyprus", "Czech Republic", "Democratic Republic of the Congo", "Denmark", "Djibouti", "Dominica", "Dominican Republic", "Ecuador", "Egypt", "El Salvador", "Equatorial Guinea", "Eritrea", "Estonia", "Eswatini", "Ethiopia", "Fiji", "Finland", "France", "Gabon", "Gambia", "Georgia", "Germany", "Ghana", "Greece", "Grenada", "Guatemala", "Guinea", "Guinea-Bissau", "Guyana", "Haiti", "Honduras", "Hungary", "Iceland", "India", "Indonesia", "Iran", "Iraq", "Ireland", "Israel", "Italy", "Ivory Coast", "Jamaica", "Japan", "Jordan", "Kazakhstan", "Kenya", "Kiribati", "Kuwait", "Kyrgyzstan", "Laos", "Latvia", "Lebanon", "Lesotho", "Liberia", "Libya", "Liechtenstein", "Lithuania", "Luxembourg", "Madagascar", "Malawi", "Malaysia", "Maldives", "Mali", "Malta", "Marshall Islands", "Mauritania", "Mauritius", "Mexico", "Micronesia", "Moldova", "Monaco", "Mongolia", "Montenegro", "Morocco", "Mozambique", "Myanmar", "Namibia", "Nauru", "Nepal", "Netherlands", "New Zealand", "Nicaragua", "Niger", "Nigeria", "North Korea", "North Macedonia", "Norway", "Oman", "Pakistan", "Palau", "Panama", "Papua New Guinea", "Paraguay", "Peru", "Philippines", "Poland", "Portugal", "Qatar", "Romania", "Russia", "Rwanda", "Saint Kitts and Nevis", "Saint Lucia", "Saint Vincent and the Grenadines", "Samoa", "San Marino", "Sao Tome and Principe", "Saudi Arabia", "Senegal", "Serbia", "Seychelles", "Sierra Leone", "Singapore", "Slovakia", "Slovenia", "Solomon Islands", "Somalia", "South Africa", "South Korea", "South Sudan", "Spain", "Sri Lanka", "Sudan", "Suriname", "Sweden", "Switzerland", "Syria", "Tajikistan", "Tanzania", "Thailand", "Timor-Leste", "Togo", "Tonga", "Trinidad and Tobago", "Tunisia", "Turkey", "Turkmenistan", "Tuvalu", "Uganda", "Ukraine", "United Arab Emirates", "United Kingdom", "United States", "Uruguay", "Uzbekistan", "Vanuatu", "Vatican City", "Venezuela", "Vietnam", "Yemen", "Zambia", "Zimbabwe"]

self.E17 = ctk.CTkComboBox(master=frame, values = Country)

self.E17.set("Country")

self.E17.grid(row=10, column=1, padx=10, pady=5,sticky="w")

button\_upload = ctk.CTkButton(master=frame, text='UPLOAD', command=self.Upload)

button\_upload.grid(row=12, column=0, padx=10, pady=5,sticky="e")

button\_clear = ctk.CTkButton(master=frame, text='CLEAR', command=self.Clear)

button\_clear.grid(row=12, column=1, padx=10, pady=5,sticky="w")

button\_update = ctk.CTkButton(master=frame, text='UPDATE', command=self.Update)

button\_update.grid(row=13, column=0, padx=10, pady=5,sticky="e")

button\_marks = ctk.CTkButton(master=frame, text='MARKS', command=self.Marks)

button\_marks.grid(row=13, column=1, padx=10, pady=5,sticky="w")

frame.grid\_columnconfigure(0, weight=1)

frame.grid\_columnconfigure(1, weight=1)

app1.mainloop()

def Clear(self):

self.E7.delete(0, END)

self.E8.delete(0, END)

self.E9.set("Gender")

self.E10.delete(0, END)

self.E11.delete(0, END)

self.E12.set("Nationality")

self.E13.delete(0, END)

self.E14.set("City")

self.E15.set("State")

self.E16.delete(0, END)

self.E17.set("Country")

tkmb.showinfo("", "Fields are cleared")

def Update(self):

try:

m9 = sqlite3.connect('career12.db')

c = m9.cursor()

x1 = self.E7.get()

z1 = self.E8.get()

h1 = self.E9.get()

i1 = self.E10.get()

j1 = self.E11.get()

k1 = self.E12.get()

l1 = self.E13.get()

m1 = self.E14.get()

n1 = self.E15.get()

o1 = self.E16.get()

p1 = self.E17.get()

if any(not field for field in (x1, z1, h1, i1, j1, k1, l1, m1, n1, o1, p1)):

tkmb.showinfo("", "Please fill all the fields")

return

c.execute("UPDATE MEMBER\_DETAILS SET FULL\_NAME=?, EMAIL=?, PHONE\_NO=?, GENDER=?, DOB=?, NATIONALITY=?,STREET\_ADDRESS=?,CITY=?, STATE=?, ZIP\_CODE=?,COUNTRY=?",

(x1, z1, h1, i1, j1, k1, l1, m1, n1, o1, p1))

m9.commit()

tkmb.showinfo("", "Member Details Updated Successfully")

except sqlite3.Error as e:

print("Error occurred:", e)

tkmb.showinfo("", "An error occurred while updating member details")

except sqlite3.IntegrityError:

print("Error occurred: UNIQUE constraint failed")

tkmb.showinfo("", "Member with the same name already exists")

finally:

c.close()

def Upload(self):

self.connector = sqlite3.connect('career12.db')

self.cursorObj = self.connector.cursor()

self.cursorObj.execute("CREATE TABLE if not exists MEMBER\_DETAILS(MEMBER\_ID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,FULL\_NAME TEXT, EMAIL TEXT, PHONE\_NO TEXT, GENDER TEXT, DOB TEXT, NATIONALITY TEXT,STREET\_ADDRESS TEXT,CITY TEXT, STATE TEXT,ZIP\_CODE TEXT,COUNTRY TEXT)")

m3 = sqlite3.connect('career12.db')

print("Connected")

x = self.E7.get()

z = self.E8.get()

h = self.E9.get()

i = self.E10.get()

j = self.E11.get()

k = self.E12.get()

l = self.E13.get()

m = self.E14.get()

n = self.E15.get()

o = self.E16.get()

p = self.E17.get()

if (x == "") or (z == "") or (h == "") or (i == "") or (j == "") or (k == "")or (l == "")or (m == "")or (n == "")or (o == "")or (p == ""):

tkmb.showinfo("", "Fill the Empty Field")

else:

try:

m3.execute("INSERT INTO MEMBER\_DETAILS(FULL\_NAME, EMAIL, PHONE\_NO, GENDER, DOB, NATIONALITY,STREET\_ADDRESS,CITY, STATE,ZIP\_CODE,COUNTRY) VALUES(?,?,?,?,?,?,?,?,?,?,?)", (x, z, h, i, j, k, l, m, n, o, p))

m3.commit()

tkmb.showinfo("","Member Details Uploaded")

except sqlite3.Error as e:

print("Error occurred:", e)

tkmb.showinfo("", "An error occurred while uploading member details")

def Marks(self):

app2 = ctk.CTk()

app2.geometry("400x500")

app2.title("12th MARKS")

label = ctk.CTkLabel(app2, text="MEMBER MARKS INFO")

label.pack(pady=20)

frame = ctk.CTkFrame(master=app2)

frame.pack(padx=40, fill='both', expand=True)

label\_name = ctk.CTkLabel(master=frame, text="TAMIL:")

label\_name.grid(row=1, column=0, padx=10, pady=5, sticky="e")

self.E26 = ctk.CTkEntry(master=frame, placeholder\_text="TAMIL")

self.E26.grid(row=1, column=1, padx=10, pady=5,sticky="w")

label\_name = ctk.CTkLabel(master=frame, text="ENGLISH:")

label\_name.grid(row=2, column=0, padx=10, pady=5, sticky="e")

self.E27 = ctk.CTkEntry(master=frame, placeholder\_text="ENGLISH")

self.E27.grid(row=2, column=1, padx=10, pady=5,sticky="w")

label\_name = ctk.CTkLabel(master=frame, text="MATHS:")

label\_name.grid(row=3, column=0, padx=10, pady=5, sticky="e")

self.E28 = ctk.CTkEntry(master=frame, placeholder\_text="MATHS")

self.E28.grid(row=3, column=1, padx=10, pady=5,sticky="w")

label\_name = ctk.CTkLabel(master=frame, text="BIOLOGY:")

label\_name.grid(row=4, column=0, padx=10, pady=5, sticky="e")

self.E29 = ctk.CTkEntry(master=frame, placeholder\_text="BIOLOGY")

self.E29.grid(row=4, column=1, padx=10, pady=5,sticky="w")

label\_name = ctk.CTkLabel(master=frame, text="PHYSICS:")

label\_name.grid(row=5, column=0, padx=10, pady=5, sticky="e")

self.E30 = ctk.CTkEntry(master=frame, placeholder\_text="PHYSICS")

self.E30.grid(row=5, column=1, padx=10, pady=5,sticky="w")

label\_name = ctk.CTkLabel(master=frame, text="CHEMISTRY:")

label\_name.grid(row=6, column=0, padx=10, pady=5, sticky="e")

self.E31 = ctk.CTkEntry(master=frame, placeholder\_text="CHEMISTRY")

self.E31.grid(row=6, column=1, padx=10, pady=5,sticky="w")

label\_name = ctk.CTkLabel(master=frame, text="TOTAL:")

label\_name.grid(row=7, column=0, padx=10, pady=5, sticky="e")

self.E32 = ctk.CTkEntry(master=frame, placeholder\_text="TOTAL")

self.E32.grid(row=7, column=1, padx=10, pady=5,sticky="w")

label\_name = ctk.CTkLabel(master=frame, text="CUTOFF:")

label\_name.grid(row=8, column=0, padx=10, pady=5, sticky="e")

self.E33 = ctk.CTkEntry(master=frame, placeholder\_text="CUTOFF")

self.E33.grid(row=8, column=1, padx=10, pady=5,sticky="w")

button\_Upload\_mark = ctk.CTkButton(master=frame, text='UPLOAD', command=self.Upload\_mark)

button\_Upload\_mark.grid(row=11, column=0, padx=10, pady=5,sticky="e")

button\_total\_mark = ctk.CTkButton(master=frame, text='TOTAL', command=self.total\_mark)

button\_total\_mark.grid(row=11, column=1, padx=10, pady=5,sticky="w")

button\_Clear\_mark = ctk.CTkButton(master=frame, text='CLEAR', command=self.Clear\_mark)

button\_Clear\_mark.grid(row=12, column=1, padx=10, pady=5,sticky="w")

button\_update\_mark = ctk.CTkButton(master=frame, text='UPDATE', command=self.update\_mark)

button\_update\_mark.grid(row=12, column=0, padx=10, pady=5,sticky="e")

button\_cutoff\_mark = ctk.CTkButton(master=frame, text='CUTOFF', command=self.cutoff\_mark)

button\_cutoff\_mark.grid(row=13, column=1, padx=10, pady=5,sticky="w")

frame.grid\_columnconfigure(0, weight=1)

frame.grid\_columnconfigure(1, weight=1)

app2.mainloop()

def Clear\_mark(self):

self.E26.delete(0,END)

self.E27.delete(0,END)

self.E28.delete(0,END)

self.E29.delete(0,END)

self.E30.delete(0,END)

self.E31.delete(0,END)

self.E32.delete(0,END)

self.E33.delete(0,END)

tkmb.showinfo("", "Fields are cleared")

def update\_mark(self):

m7 = sqlite3.connect('career12.db')

c1 = m7.cursor()

z3 = self.E26.get()

h3 = self.E27.get()

i3 = self.E28.get()

j3 = self.E29.get()

k3 = self.E30.get()

l3 = self.E31.get()

m3 = self.E32.get()

n3 = self.E33.get()

if any(not field for field in (z3, h3, i3, j3, k3,l3,m3,n3)):

tkmb.showinfo("", "Please fill all the fields")

return

try:

c1.execute("UPDATE MEMBER\_MARKS SET TAMIL=?, ENGLISH=?, MATHS=?, BIOLOGY=?, PHYSICS=?,CHEMISTRY =?,TOTAL=?,CUTOFF=?",(z3, h3, i3, j3, k3,l3,m3,n3))

m7.commit()

tkmb.showinfo("", "Member Marks Updated Successfully")

except sqlite3.Error as e:

print("Error occurred:", e)

tkmb.showinfo("", "An error occurred while updating member marks")

except sqlite3.IntegrityError:

print("Error occurred: UNIQUE constraint failed")

tkmb.showinfo("", "Member with the same student id already exists")

finally:

m7.close()

def Upload\_mark(self):

self.connector = sqlite3.connect('career12.db')

self.cursorObj = self.connector.cursor()

self.cursorObj.execute("CREATE TABLE if not exists MEMBER\_MARKS(STUDENT\_ID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,TAMIL INTEGER,ENGLISH INTEGER,MATHS INTEGER,BIOLOGY INTEGER,PHYSICS INTEGER,CHEMISTRY INTEGER,TOTAL INTEGER,CUTOFF INTEGER)")

m5 = sqlite3.connect('career12.db')

print("Connected")

z4 = self.E26.get()

h4 = self.E27.get()

i4 = self.E28.get()

j4 = self.E29.get()

k4 = self.E30.get()

l4 = self.E31.get()

m4 = self.E32.get()

n4 = self.E33.get()

if (z4 == "") or (h4 == "") or (i4 == "") or (j4 == "") or (k4 == "")or (l4 == "")or (m4 == "")or (n4 == ""):

tkmb.showinfo("", "Fill the Empty Field")

else:

try:

m5.execute("INSERT INTO MEMBER\_MARKS(TAMIL ,ENGLISH ,MATHS ,BIOLOGY ,PHYSICS ,CHEMISTRY,TOTAL,CUTOFF) VALUES(?,?,?,?,?,?,?,?)", (z4, h4, i4, j4, k4, l4, m4, n4,))

m5.commit()

tkmb.showinfo("","Member Marks Uploaded")

except sqlite3.Error as e:

print("Error occurred:", e)

tkmb.showinfo("", "An error occurred while uploading member marks")

def total\_mark(self):

try:

z5 = int(self.E26.get())

h5 = int(self.E27.get())

i5 = int(self.E28.get())

j5 = int(self.E29.get())

k5 = int(self.E30.get())

l5 = int(self.E31.get())

tot = (z5 + h5 + i5 + j5 + k5 + l5) / 600 \* 100

self.E32.delete(0, ctk.END)

self.E32.insert(0, f"{tot:.2f}%")

except ValueError:

tkmb.showinfo("", "Please enter valid numbers for all marks")

def cutoff\_mark(self):

try:

i6 = int(self.E28.get())

k6 = int(self.E30.get())

l6 = int(self.E31.get())

avg = i6 + (k6 / 2) + (l6 / 2)

self.E33.delete(0, ctk.END)

self.E33.insert(0, f"{avg:.2f}")

except ValueError:

tkmb.showinfo("", "Please enter valid numbers for all marks")

app = App()

app.mainloop()