import customtkinter as tk

from tkinter import \*

from tkinter import ttk

import mysql.connector

import tkinter.messagebox as tkmb

import docx

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

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

con = mysql.connector.connect(

host="localhost", user="root", password="Naren", database="new\_project"

)

cur = con.cursor()

def log\_event(event):

q = "select tid,tpass from teacher"

cur.execute(q)

tdata = cur.fetchall()

tid = [i[0] for i in tdata]

tpass = [i[1] for i in tdata]

q1 = "select sid,spass from studentbio"

cur.execute(q1)

sdata = cur.fetchall()

sid = [i[0] for i in sdata]

spass = [i[1] for i in sdata]

if loginpass.get() == "admin":

loginwin.destroy()

admin()

elif int(loginid.get()) in tid:

index = tid.index(int(loginid.get()))

if loginpass.get() == tpass[index]:

a = loginid.get()

loginwin.destroy()

teacher(a)

else:

tkmb.showwarning("Invaild", "Wrong Password")

elif int(loginid.get()) in sid:

index = sid.index(int(loginid.get()))

if loginpass.get() == spass[index]:

a = loginid.get()

loginwin.destroy()

student(a)

else:

tkmb.showwarning("Invaild", "Wrong Password")

else:

tkmb.showwarning("Invail ID", "ID not Found")

def log():

q = "select tid,tpass from teacher"

cur.execute(q)

tdata = cur.fetchall()

tid = [i[0] for i in tdata]

tpass = [i[1] for i in tdata]

q1 = "select sid,spass from studentbio"

cur.execute(q1)

sdata = cur.fetchall()

sid = [i[0] for i in sdata]

spass = [i[1] for i in sdata]

if loginpass.get() == "admin":

loginwin.destroy()

admin()

elif int(loginid.get()) in tid:

index = tid.index(int(loginid.get()))

if loginpass.get() == tpass[index]:

a = loginid.get()

loginwin.destroy()

teacher(a)

else:

tkmb.showwarning("Invaild", "Wrong Password")

elif int(loginid.get()) in sid:

index = sid.index(int(loginid.get()))

if loginpass.get() == spass[index]:

a = loginid.get()

loginwin.destroy()

student(a)

else:

tkmb.showwarning("Invaild", "Wrong Password")

else:

tkmb.showwarning("Invail ID", "ID Not Found")

def admin():

adwin = tk.CTk()

adwin.title("Admin")

adwin.geometry("1300x700+0+0")

adtab = tk.CTkTabview(adwin, width=1300, height=650)

adtab.pack(padx=20, pady=20)

distea = adtab.add("Display Teacher")

distub = adtab.add("Display Student Bio")

# distum = adtab.add("Display Student mark")

adstub = adtab.add("Add Student bio")

adtea = adtab.add("Add Teacher")

astea = adtab.add("Assign class For teacher")

# display teacher

s = ttk.Style(distea)

s.theme\_use('clam')

s.configure('Treeview',rowheight=40)

tree1 = ttk.Treeview(distea, height=600,show='headings')

tree1.pack(padx=10, pady=10)

tree1["columns"] = ("tid", "tname", "classt", "tclasses")

tree1.column("#0", width=0, anchor="center")

tree1.column("tid", width=100, anchor="center")

tree1.column("tname", width=200, anchor="center")

tree1.column("classt", width=100, anchor="center")

tree1.column("tclasses", width=200, anchor="center")

tree1.heading("tid", text="Teacher\_Id")

tree1.heading("tname", text="Teacher\_Name")

tree1.heading("classt", text="Class teacher of")

tree1.heading("tclasses", text="Handling Classes")

q1 = "select tid,tname,class\_teacher,handling\_classes from teacher"

cur.execute(q1)

tdata = cur.fetchall()

for i in range(len(tdata)):

tree1.insert(parent="", index="end", iid=i, values=tdata[i])

# dispay studentbio

def getstub(x):

cur = con.cursor()

if x == "All":

sql = f"select sid,sname,sclass,dob,fname,mname from studentbio"

else:

sql = f"select sid,sname,sclass,dob,fname,mname from studentbio where sclass = '{x}'"

cur.execute(sql)

return cur.fetchall()

def changestub(event):

for j in tree2.get\_children():

tree2.delete(j)

inval = getstub(combo.get())

for i in range(len(inval)):

tree2.insert(parent="", index="end", iid=i, values=inval[i])

def changein():

for j in tree2.get\_children():

tree2.delete(j)

inval = getstub(combo.get())

for i in range(len(inval)):

tree2.insert(parent="", index="end", iid=i, values=inval[i])

val = getval()

val.insert(0, "All")

combo = ttk.Combobox(distub, width=20, values=val, state="readonly")

combo.pack(pady=10)

combo.bind("<<ComboboxSelected>>", changestub)

combo.set("All")

s = ttk.Style(distub)

s.theme\_use('clam')

s.configure('Treeview',rowheight=40)

tree2 = ttk.Treeview(distub,show='headings', height=600)

tree2.pack(padx=10, pady=10)

tree2["columns"] = ("sid", "sname", "sclass", "dob", "fname", "mname")

tree2.column("#0", width=0, anchor="center")

tree2.column("sid", width=100, anchor="center")

tree2.column("sname", width=200, anchor="center")

tree2.column("sclass", width=100, anchor="center")

tree2.column("dob", width=100, anchor="center")

tree2.column("fname", width=200, anchor="center")

tree2.column("mname", width=200, anchor="center")

tree2.heading("sid", text="Admission\_Id")

tree2.heading("sname", text="Student\_Name")

tree2.heading("sclass", text="Student\_Class")

tree2.heading("dob", text="Date of Birth")

tree2.heading("fname", text="Father\_Name")

tree2.heading("mname", text="Mother\_Name")

q2 = "select sid,sname,sclass ,dob ,fname,mname from studentbio"

cur.execute(q2)

sdata = cur.fetchall()

for i in range(len(sdata)):

tree2.insert(parent="", index="end", iid=i, values=sdata[i])

# add student bio

inframe = tk.CTkFrame(adstub)

inframe.pack()

adidlab = tk.CTkLabel(inframe, text="Admn\_Id", font=("Arial", 24))

adidlab.grid(row=0, column=0, padx=20, pady=30)

adiden = tk.CTkEntry(

inframe,

placeholder\_text="Enter the Admission ID of Student",

width=210,

height=32,

)

adiden.grid(row=0, column=1, padx=20, pady=30)

namelab = tk.CTkLabel(inframe, text="Name", font=("Arial", 24))

namelab.grid(row=0, column=3, padx=20, pady=30)

namen = tk.CTkEntry(

inframe, placeholder\_text="Enter the Name of Student", width=200, height=32

)

namen.grid(row=0, column=4, padx=20, pady=30)

class\_seclab = tk.CTkLabel(inframe, text="Class\_Sec", font=("Arial", 24))

class\_seclab.grid(row=1, column=0, padx=20, pady=30)

classcombo = ttk.Combobox(inframe, width=20, values=getval(), state="readonly")

classcombo.grid(row=1, column=1, padx=20, pady=30)

doblab = tk.CTkLabel(inframe, text="Dob", font=("Arial", 24))

doblab.grid(row=1, column=3, padx=20, pady=30)

doben = tk.CTkEntry(

inframe, placeholder\_text="Enter the Dob of Student", width=200, height=32

)

doben.grid(row=1, column=4, padx=20, pady=30)

fnamelab = tk.CTkLabel(inframe, text="Father Name", font=("Arial", 24))

fnamelab.grid(row=2, column=0, padx=20, pady=30)

fnamen = tk.CTkEntry(

inframe,

placeholder\_text="Enter the Father Name of Student",

width=200,

height=32,

)

fnamen.grid(row=2, column=1, padx=20, pady=30)

mnamelab = tk.CTkLabel(inframe, text="Mother Name", font=("Arial", 24))

mnamelab.grid(row=2, column=3, padx=20, pady=30)

mnamen = tk.CTkEntry(

inframe,

placeholder\_text="Enter the Mother Name of Student",

width=210,

height=32,

)

mnamen.grid(row=2, column=4, padx=20, pady=30)

spassl = tk.CTkLabel(inframe, text="Password", font=("Arial", 24))

spassl.grid(row=3, column=0)

spassn = tk.CTkEntry(

inframe,

placeholder\_text="Enter the password of the student",

width=260,

height=32,

)

spassn.grid(row=3, column=1)

def submit():

admn = adiden.get() or "NULL"

name = namen.get() or "NULL"

class\_sec = classcombo.get() or "NULL"

dob = doben.get() or "NULL"

fname = fnamen.get() or "NUll"

mname = mnamen.get() or "NULL"

spass = spassn.get()

cur = con.cursor()

sql = f"insert into studentbio values({admn},'{spass}','{name}','{class\_sec}','{dob}','{fname}','{mname}')"

cur.execute(sql)

con.commit()

sql2 =f"insert into studentmark(sid,sclass) values({admn},'{class\_sec}')"

cur.execute(sql2)

con.commit()

adiden.delete(0, END)

namen.delete(0, END)

doben.delete(0, END)

fnamen.delete(0, END)

mnamen.delete(0, END)

spassn.delete(0, END)

tkmb.showinfo("Insert", "Inserted Succesfully")

adtab.set("Display Student Bio")

changein()

getbut = tk.CTkButton(adstub, text="Submit", command=submit)

getbut.pack(pady=30)

# add teacher

def gettb():

cur = con.cursor()

q = "select tid,tname,class\_teacher,handling\_classes from teacher"

cur.execute(q)

return cur.fetchall()

def tchangein():

for j in tree1.get\_children():

tree1.delete(j)

inval = gettb()

for i in range(len(inval)):

tree1.insert(parent="", index="end", iid=i, values=inval[i])

inframe1 = tk.CTkFrame(adtea)

inframe1.pack()

tidlab = tk.CTkLabel(inframe1, text="Teacher\_Id", font=("Arial", 24))

tidlab.grid(row=0, column=0, padx=20, pady=30)

tiden = tk.CTkEntry(

inframe1,

placeholder\_text="Enter the teacher ID of the Teacher",

width=210,

height=32,

)

tiden.grid(row=0, column=1, padx=20, pady=30)

tnamelab = tk.CTkLabel(inframe1, text="Name", font=("Arial", 24))

tnamelab.grid(row=0, column=3, padx=20, pady=30)

tnamen = tk.CTkEntry(

inframe1, placeholder\_text="Enter the Name of the Teacher", width=200, height=32

)

tnamen.grid(row=0, column=4, padx=20, pady=30)

class\_teacherlab = tk.CTkLabel(

inframe1, text="Class\_Teacher\_of", font=("Arial", 24)

)

class\_teacherlab.grid(row=1, column=0, padx=20, pady=30)

tclasscombo = ttk.Combobox(inframe1, width=20, values=getval(), state="readonly")

tclasscombo.grid(row=1, column=1, padx=20, pady=30)

handling\_class = tk.CTkLabel(inframe1, text="Handling\_Classes", font=("Arial", 24))

handling\_class.grid(row=1, column=3, padx=20, pady=30)

hanen = tk.CTkEntry(

inframe1, placeholder\_text="Enter the Handling Classes", width=200, height=32

)

hanen.grid(row=1, column=4, padx=20, pady=30)

tpasslab = tk.CTkLabel(inframe1, text="Teacher Password", font=("Arial", 24))

tpasslab.grid(row=2, column=0, padx=20, pady=30)

tpassen = tk.CTkEntry(

inframe1,

placeholder\_text="Enter the Password of Teacher",

width=200,

height=32,

)

tpassen.grid(row=2, column=1, padx=20, pady=30)

def tsubmit():

tid = tiden.get() or "NULL"

tname = tnamen.get() or "NULL"

classt = tclasscombo.get() or "NULL"

hanclass = hanen.get() or "NULL"

tpass1 = tpassen.get() or "NUll"

cur = con.cursor()

sql = f"insert into teacher values({tid},'{tpass1}','{tname}','{classt}','{hanclass}')"

cur.execute(sql)

con.commit()

tiden.delete(0, END)

tnamen.delete(0, END)

hanen.delete(0, END)

tpassen.delete(0, END)

tkmb.showinfo("Insert", "Inserted Succesfully")

adtab.set("Display Teacher")

tchangein()

getbut2 = tk.CTkButton(adtea, text="Submit", command=tsubmit)

getbut2.pack(pady=30)

# assign a class for teacher

atframe = tk.CTkFrame(astea)

atframe.pack()

astidlab = tk.CTkLabel(atframe, text="Teacher\_Id", font=("Arial", 24))

astidlab.grid(row=0, column=0, padx=20, pady=30)

astiden = tk.CTkEntry(

atframe,

placeholder\_text="Enter the teacher ID of the Teacher",

width=220,

height=32,

)

astiden.grid(row=0, column=1, padx=20, pady=30)

asclasslab = tk.CTkLabel(atframe, text="Handling\_Classes", font=("Arial", 24))

asclasslab.grid(row=1, column=0, padx=20, pady=30)

asen = tk.CTkEntry(

atframe, placeholder\_text="Enter the Classes to assign", width=200, height=32

)

asen.grid(row=1, column=1, padx=20, pady=30)

def updateclass():

astid = astiden.get()

asclass = asen.get()

asclass = "," + asclass

q = f"update teacher set handling\_classes = concat(handling\_classes,'{asclass}') where tid={astid}"

cur = con.cursor()

cur.execute(q)

con.commit()

astiden.delete(0, END)

asen.delete(0, END)

tkmb.showinfo("Update", "Updated Succesfully")

adtab.set("Display Teacher")

tchangein()

getbut3 = tk.CTkButton(astea, text="Submit", command=updateclass)

getbut3.pack(pady=30)

adwin.mainloop()

def teacher(tid):

def t\_mksadd():

def change\_marks():

examval = exam.get()

entryval = entry.get()

markval = mark.get()

q = "update studentmark set {} = {} where sid = {}".format(

examval, markval, entryval

)

cur.execute(q)

con.commit()

donewindow\_messagebox = tkmb.showinfo(

"Done!", "The record has been updated!"

)

mksadd = tk.CTk()

mksadd.geometry("800x600")

mksadd.resizable(width=False, height=False)

mksadd.title("Modify Marks")

entry = tk.CTkEntry(mksadd, width=200, placeholder\_text="Enter the roll number")

entry.focus()

entry.pack()

exam = ttk.Combobox(

mksadd,

width=30,

values=[

"ut1\_sub1",

"ut1\_sub2",

"ut1\_sub3",

"ut1\_sub4",

"ut1\_sub5",

"ut2\_sub1",

"ut2\_sub2",

"ut2\_sub3",

"ut2\_sub4",

"ut2\_sub5",

"ut3\_sub1",

"ut3\_sub2",

"ut3\_sub3",

"ut3\_sub4",

"ut3\_sub5",

"qt1\_sub1",

"qt1\_sub2",

"qt1\_sub3",

"qt1\_sub4",

"qt1\_sub5",

"ut4\_sub1",

"ut4\_sub2",

"ut4\_sub3",

"ut4\_sub4",

"ut4\_sub5",

"ut5\_sub1",

"ut5\_sub2",

"ut5\_sub3",

"ut5\_sub4",

"ut5\_sub5",

"ht1\_sub1",

"ht1\_sub2",

"ht1\_sub3",

"ht1\_sub4",

"ht1\_sub5",

"at1\_sub1",

"at1\_sub2",

"at1\_sub3",

"at1\_sub4",

"at1\_sub5",

],

state="readonly",

)

exam.pack(padx=5, pady=5)

exam.set("Select test")

values = [

"ut1\_sub1",

"ut1\_sub2",

"ut1\_sub3",

"ut1\_sub4",

"ut1\_sub5",

"ut2\_sub1",

"ut2\_sub2",

"ut2\_sub3",

"ut2\_sub4",

"ut2\_sub5",

"ut3\_sub1",

"ut3\_sub2",

"ut3\_sub3",

"ut3\_sub4",

"ut3\_sub5",

"qt1\_sub1",

"qt1\_sub2",

"qt1\_sub3",

"qt1\_sub4",

"qt1\_sub5",

"ut4\_sub1",

"ut4\_sub2",

"ut4\_sub3",

"ut4\_sub4",

"ut4\_sub5",

"ut5\_sub1",

"ut5\_sub2",

"ut5\_sub3",

"ut5\_sub4",

"ut5\_sub5",

"ht1\_sub1",

"ht1\_sub2",

"ht1\_sub3",

"ht1\_sub4",

"ht1\_sub5",

"at1\_sub1",

"at1\_sub2",

"at1\_sub3",

"at1\_sub4",

"at1\_sub5",

]

mark = tk.CTkEntry(mksadd, width=200, placeholder\_text="Enter mark :")

mark.focus()

mark.pack()

dobut = tk.CTkButton(

mksadd, width=200, text="Click to do the changes", command=change\_marks

)

dobut.pack()

mksadd.mainloop()

def generate():

def dummyfunc():

id1 = stuid.get()

q1 = f"select \* from studentbio where sid = {id1}"

cur.execute(q1)

data1 = cur.fetchall()

q2 = f"select \* from studentmark where sid = {id1}"

cur.execute(q2)

data2 = cur.fetchall()

doc = docx.Document()

doc.add\_heading("Report Card", 0)

doc.add\_heading("BIODATA", level=1)

doc.add\_paragraph(f"Father name : {data1[0][5]}")

doc.add\_paragraph(f"Mother's name : {data1[0][6]}")

doc.add\_paragraph(f"Student Name : {data1[0][2]}")

doc.add\_paragraph(f"Class and Section : {data1[0][3]}")

doc.add\_paragraph(f"Date Of Birth : {data1[0][4]}")

doc.add\_page\_break()

doc.add\_heading("Marks", level=1)

record = data1[0]

tab = doc.add\_table(rows=1, cols=9)

tab.style = "Colorful List"

header\_cell = tab.rows[0].cells

header\_cell[0].text = "Subject"

header\_cell[1].text = "UT-1"

header\_cell[2].text = "UT-2"

header\_cell[3].text = "UT-3"

header\_cell[4].text = "QT"

header\_cell[5].text = "UT-4"

header\_cell[6].text = "UT-5"

header\_cell[7].text = "HT"

header\_cell[8].text = "AT"

for i in data2: # [(12128,12A,ut1\_sub1........)]

ut1tot = (

(i[2] or 0) + (i[3] or 0) + (i[4] or 0) + (i[5] or 0) + (i[6] or 0)

)

ut2tot = (

(i[7] or 0)

+ (i[8] or 0)

+ (i[9] or 0)

+ (i[10] or 0)

+ (i[11] or 0)

)

ut3tot = (

(i[12] or 0)

+ (i[13] or 0)

+ (i[14] or 0)

+ (i[15] or 0)

+ (i[16] or 0)

)

quatot = (

(i[17] or 0)

+ (i[18] or 0)

+ (i[19] or 0)

+ (i[20] or 0)

+ (i[21] or 0)

)

ut4tot = (

(i[22] or 0)

+ (i[23] or 0)

+ (i[24] or 0)

+ (i[25] or 0)

+ (i[26] or 0)

)

ut5tot = (

(i[27] or 0)

+ (i[28] or 0)

+ (i[29] or 0)

+ (i[30] or 0)

+ (i[31] or 0)

)

hattot = (

(i[32] or 0)

+ (i[33] or 0)

+ (i[34] or 0)

+ (i[35] or 0)

+ (i[36] or 0)

)

auttot = (

(i[37] or 0)

+ (i[38] or 0)

+ (i[39] or 0)

+ (i[40] or 0)

+ (i[41] or 0)

)

for j in range(0, 6):

row\_cells = tab.add\_row().cells

if j == 0:

row\_cells[0].text = "Sub1"

row\_cells[1].text = str(i[2]) or 0

row\_cells[2].text = str(i[7]) or 0

row\_cells[3].text = str(i[12]) or 0

row\_cells[4].text = str(i[17]) or 0

row\_cells[5].text = str(i[22]) or 0

row\_cells[6].text = str(i[27]) or 0

row\_cells[7].text = str(i[32]) or 0

row\_cells[8].text = str(i[37]) or 0

elif j == 1:

row\_cells[0].text = "Sub2"

row\_cells[1].text = str(i[3]) or 0

row\_cells[2].text = str(i[8]) or 0

row\_cells[3].text = str(i[13]) or 0

row\_cells[4].text = str(i[18]) or 0

row\_cells[5].text = str(i[23]) or 0

row\_cells[6].text = str(i[28]) or 0

row\_cells[7].text = str(i[33]) or 0

row\_cells[8].text = str(i[38]) or 0

elif j == 2:

row\_cells[0].text = "Sub3"

row\_cells[1].text = str(i[4]) or 0

row\_cells[2].text = str(i[9]) or 0

row\_cells[3].text = str(i[14]) or 0

row\_cells[4].text = str(i[19]) or 0

row\_cells[5].text = str(i[24]) or 0

row\_cells[6].text = str(i[29]) or 0

row\_cells[7].text = str(i[34]) or 0

row\_cells[8].text = str(i[39]) or 0

elif j == 3:

row\_cells[0].text = "Sub4"

row\_cells[1].text = str(i[5]) or 0

row\_cells[2].text = str(i[10]) or 0

row\_cells[3].text = str(i[15]) or 0

row\_cells[4].text = str(i[20]) or 0

row\_cells[5].text = str(i[25]) or 0

row\_cells[6].text = str(i[30]) or 0

row\_cells[7].text = str(i[35]) or 0

row\_cells[8].text = str(i[40]) or 0

elif j == 4:

row\_cells[0].text = "Sub5"

row\_cells[1].text = str(i[6]) or 0

row\_cells[2].text = str(i[11]) or 0

row\_cells[3].text = str(i[16]) or 0

row\_cells[4].text = str(i[21]) or 0

row\_cells[5].text = str(i[26]) or 0

row\_cells[6].text = str(i[31]) or 0

row\_cells[7].text = str(i[36]) or 0

row\_cells[8].text = str(i[41]) or 0

elif j == 5:

row\_cells[0].text = "Total"

row\_cells[1].text = str(ut1tot) or 0

row\_cells[2].text = str(ut2tot) or 0

row\_cells[3].text = str(ut3tot) or 0

row\_cells[4].text = str(quatot) or 0

row\_cells[5].text = str(ut4tot) or 0

row\_cells[6].text = str(ut5tot) or 0

row\_cells[7].text = str(hattot) or 0

row\_cells[8].text = str(auttot) or 0

doc.save(f"{id1}.docx")

tkmb.showinfo("Done!", "Saved!")

gwin = tk.CTk()

gwin.title("Generate a report card")

stuid = tk.CTkEntry(gwin, placeholder\_text="Enter the student id")

stuid.pack(padx=20, pady=30)

submitb = tk.CTkButton(gwin, text="Submit", command=dummyfunc)

submitb.pack(padx=20, pady=30)

gwin.mainloop()

twin = tk.CTk()

twin.state("zoomed")

twin.geometry("1300x700+0+0")

tcode = tk.CTkLabel(twin, text=f"Teacher Code\n{tid}", font=("Arial", 25))

tcode.place(x=100, y=300)

option\_addmks = tk.CTkButton(

master=twin, text="Modify Marks", command=t\_mksadd, hover=True

)

option\_addmks.place(x=300, y=300)

option\_generaterpcard = tk.CTkButton(

master=twin, text="Generate a report card", command=generate

)

option\_generaterpcard.place(x=450, y=300)

twin.title(tid)

twin.mainloop()

def student(sid):

def display\_button():

q = f"select \* from studentmark where sid = {sid}"

cur.execute(q)

dat12 = cur.fetchall()

newin = tk.CTk()

newin.title('Display Marks')

newin.geometry("700x700")

s = ttk.Style(newin)

s.theme\_use('clam')

s.configure('Treeview',rowheight=40)

tree3 = ttk.Treeview(newin, show="headings", height=7)

tree3.pack(padx=20, pady=30)

tree3["columns"] = (

"sub\\test",

"ut1",

"ut2",

"ut3",

"quarterly",

"ut4",

"halfyearly",

"ut5",

"annualexam",

)

tree3.column("#0", width=-1, anchor="center")

tree3.column("sub\\test", width=100, anchor="center")

tree3.column("ut1", width=50, anchor="center")

tree3.column("ut2", width=50, anchor="center")

tree3.column("ut3", width=50, anchor="center")

tree3.column("quarterly", width=100, anchor="center")

tree3.column("ut4", width=50, anchor="center")

tree3.column("halfyearly", width=100, anchor="center")

tree3.column("ut5", width=50, anchor="center")

tree3.column("annualexam", width=100, anchor="center")

tree3.heading("sub\\test", text="Sub\\Test")

tree3.heading("ut1", text="UT-1")

tree3.heading("ut2", text="UT-2")

tree3.heading("ut3", text="UT-3")

tree3.heading("quarterly", text="Quarterly")

tree3.heading("ut4", text="UT-4")

tree3.heading("halfyearly", text="Half-Yearly")

tree3.heading("ut5", text="UT-5")

tree3.heading("annualexam", text="Annual")

for i in dat12:

for j in range(0, 6):

if j == 0:

val = ["Sub1", i[2], i[7], i[12], i[17], i[22], i[27], i[32], i[37]]

tree3.insert(parent="", index="end", iid=j, values=val)

if j == 1:

val = ["Sub2", i[3], i[8], i[13], i[18], i[23], i[28], i[33], i[38]]

tree3.insert(parent="", index="end", iid=j, values=val)

if j == 2:

val = ["Sub3", i[4], i[9], i[14], i[19], i[24], i[29], i[34], i[39]]

tree3.insert(parent="", index="end", iid=j, values=val)

if j == 3:

val = [

"Sub4",

i[5],

i[10],

i[15],

i[20],

i[25],

i[30],

i[35],

i[40],

]

tree3.insert(parent="", index="end", iid=j, values=val)

if j == 4:

val = [

"Sub5",

i[6],

i[11],

i[16],

i[21],

i[26],

i[31],

i[36],

i[41],

]

tree3.insert(parent="", index="end", iid=j, values=val)

if j == 5:

ut1tot = (

(i[2] or 0)

+ (i[3] or 0)

+ (i[4] or 0)

+ (i[5] or 0)

+ (i[6] or 0)

)

ut2tot = (

(i[7] or 0)

+ (i[8] or 0)

+ (i[9] or 0)

+ (i[10] or 0)

+ (i[11] or 0)

)

ut3tot = (

(i[12] or 0)

+ (i[13] or 0)

+ (i[14] or 0)

+ (i[15] or 0)

+ (i[16] or 0)

)

quatot = (

(i[17] or 0)

+ (i[18] or 0)

+ (i[19] or 0)

+ (i[20] or 0)

+ (i[21] or 0)

)

ut4tot = (

(i[22] or 0)

+ (i[23] or 0)

+ (i[24] or 0)

+ (i[25] or 0)

+ (i[26] or 0)

)

ut5tot = (

(i[27] or 0)

+ (i[28] or 0)

+ (i[29] or 0)

+ (i[30] or 0)

+ (i[31] or 0)

)

hattot = (

(i[32] or 0)

+ (i[33] or 0)

+ (i[34] or 0)

+ (i[35] or 0)

+ (i[36] or 0)

)

auttot = (

(i[37] or 0)

+ (i[38] or 0)

+ (i[39] or 0)

+ (i[40] or 0)

+ (i[41] or 0)

)

val = [

"Total",

ut1tot,

ut2tot,

ut3tot,

quatot,

ut4tot,

ut5tot,

hattot,

auttot,

]

tree3.insert(parent="", index="end", iid=j, values=val)

newin.mainloop()

swin = tk.CTk()

swin.geometry("500x500")

swin.title(sid)

slabel = tk.CTkLabel(swin,text='Student Code',font=('Arial',20))

slabel.place(x=200,y=270)

slabel2 = tk.CTkLabel(swin,text=sid,font=('Arial',18))

slabel2.place(x=230,y=300)

disp\_mark = tk.CTkButton(

swin, text="Display Marks", width=100, command=display\_button

)

disp\_mark.place(x=350, y=300)

swin.mainloop()

def getval():

res = []

for i in range(1, 13):

for j in range(65, 70):

res.append(f"{i}" f"{chr(j)}")

return res

loginwin = tk.CTk()

loginwin.geometry("450x350+500+200")

loginwin.title("Login Window")

def x(event):

loginid.focus\_set()

def y(event):

loginpass.focus\_set()

logframe = tk.CTkFrame(loginwin, width=400, height=300, fg\_color="#242424")

logframe.pack()

label1 = tk.CTkLabel(logframe, text=" Login", font=("Arial", 35))

label1.grid(row=0, column=0, padx=30, pady=30)

loginid = tk.CTkEntry(logframe, placeholder\_text="Enter your Id", width=250)

loginid.grid(row=1, column=0, padx=10, pady=20)

loginid.bind("<Enter>", x)

loginid.bind("<Return>", lambda x: loginpass.focus\_set())

loginpass = tk.CTkEntry(

logframe, placeholder\_text="Enter your password", width=250, show="\*"

)

loginpass.grid(row=2, column=0, padx=10, pady=10)

loginpass.bind("<Return>", command=log\_event)

loginpass.bind("<Enter>", y)

logbutton = tk.CTkButton(logframe, text="Login", command=log)

logbutton.grid(row=3, column=0, padx=20, pady=20)

def showpass1(event):

global pass1

if pass1 == 0:

loginpass.configure(show="")

pass1 = 1

else:

loginpass.configure(show="\*")

pass1 = 0

pass1 = 0

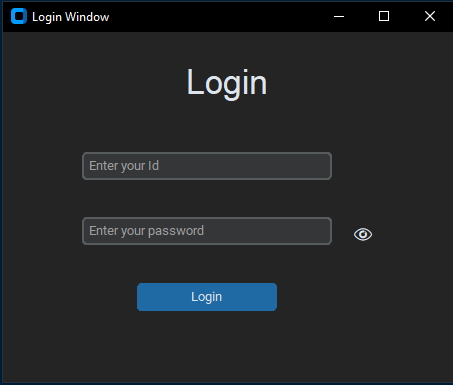
eyebutton = tk.CTkLabel(logframe, text="👁", width=20, height=20, font=("Arial", 20))

eyebutton.grid(row=2, column=1, padx=10, pady=20)

eyebutton.bind("<Enter>", showpass1)

eyebutton.bind("<Leave>", showpass1)

loginwin.mainloop()

**Output:**

