*# imports-----------------------------------------------------***import** os  
**import** customtkinter **as** css  
**import** screen\_brightness\_control **as** sbc  
**import** datetime  
**import** sqlite3  
**from** tkinter **import** messagebox  
**from** plyer **import** notification  
**from** functions\_m **import** deadline  
**from** functions\_m **import** mill\_minutes  
  
*#colour variables to use throughout the Application*background\_colour=**"#017bf5"**fore\_colour=**"#017bf5"**light=**"gray"**connection = sqlite3.connect(**"dope.db"**)  
cursor = connection.cursor()  
cursor.execute(  
 **'''create table IF NOT EXISTS deadline(courseName text,Deadline\_date text,SetDate text,remindermin   
 integer)'''**)  
  
  
*# GLOBALS-----------------------------------------------------*j=1  
d=j  
s=j  
  
  
  
  
*# dayslefts = deadline(20, 5)  
# # to remind the user throughout the day  
# # remind\_hr=mill\_minutes(int(input("reminding time in hours :")))  
# remind\_min = mill\_minutes(int(input("reminding time in minutes :")))  
# print(remind\_min)  
  
# GLOBALS-----------------------------------------------------*year=**None**day=**None**month=**None***# FUNCTION-----------------------------------------------------***def** display():  
 display\_frame.grid()  
 display\_frame.grid\_propagate(**False**)  
 others\_frame.grid\_forget()  
 coursework\_frame.grid\_forget()  
 power\_btn.configure(fg\_color=**"#017bf5"**)  
 display\_btn.configure(fg\_color=**"#434344"**)  
 coursework\_btn.configure(fg\_color=**"#017bf5"**)  
  
  
*# changing brightness of the screen***def** change\_brightness(value):  
 right\_frame.update()  
 sbc.set\_brightness(round(value))  
 *# print(value)  
  
  
# changing the gamma of the screen***def** change\_sharpness(value):  
 right\_frame.update()  
 sbc.fade\_brightness(round(value))  
  
  
*# COURSE WORK BTN  
#-----------------------------------------------------------------------------------------------------------------------***def** coursework():  
 coursework\_frame.grid()  
 coursework\_frame.grid\_propagate(**False**)  
 coursework\_btn.configure(fg\_color=**"#434344"**)  
 display\_frame.grid\_forget()  
 others\_frame.grid\_forget()  
 power\_btn.configure(fg\_color=**"#017bf5"**)  
 display\_btn.configure(fg\_color=**"#017bf5"**)  
 about\_us\_btn.configure(fg\_color=**"#017bf5"**)  
  
  
*# >>>>>>>>>>>>>>>>>>>>>>  
# A function to pick all details from a coursework pad1 and  
#-----------------------------------------------------------------------------------------------------------------------***def** getDeadlineDetailsPad1():  
 data\_view\_section.update()  
 count=0  
  
  
 *# accessing the datetime module* x = datetime.datetime.now()  
  
 current\_year = int(x.strftime(**"%Y"**))  
 current\_month = int(x.strftime(**"%m"**))  
 current\_day = int(x.strftime(**"%d"**))  
  
 *# Getting user input(ALL details of the deadline a)  
 # using N will be used to verify if the deadline is not less than current day(AVOIDING NEGATIVE VALUES)* course = courseUnitInput.get()  
  
 year = deadline\_yearInput.get()  
  
  
  
 month = deadline\_monthInput.get()  
  
  
  
 day = deadline\_dayInput.get()  
  
  
  
 reminder\_min1 = reminder\_minute\_Input.get()  
  
 *# ERROR HANDLING>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>  
  
 # error handling deadline cant be less than the current day* **if** course **and** year **and** month **and** day **and** reminder\_min1:  
 **try**:  
 *# adding a number to reminder time to avoid collision* reminder\_min = int(reminder\_min1) + 2  
 *# converting the current day to string format>>>>>>>>>>* current\_date = str(current\_year) + **"/"** + str(current\_month) + **"/"** + str(current\_day)  
 deadline\_date = year + **"/"** + month + **"/"** + day  
 *# coverting to date format>>>>>>>>>>>>>>>>>>>>>>* deadl\_fin = x.strptime(deadline\_date, **"%Y/%m/%d"**)  
  
 current\_datefin = x.strptime(current\_date, **"%Y/%m/%d"**)  
  
 *# getting days left after subtracting the current date from the deadline* daysleft = deadl\_fin - current\_datefin  
  
 print(daysleft.days - 1)  
  
 *# DATABASE CREATION----------------------------------------------* cursor.execute(  
 **'''create table IF NOT EXISTS deadline(courseName text,Deadline\_date text,SetDate text,remindermin   
 integer)'''**)  
 cursor.execute(**"insert into deadline values(:courseName,:Deadline\_date,:SetDate,:remindermin)"**,  
 {**'courseName'**: course, **'Deadline\_date'**: deadline\_date, **'SetDate'**: current\_date,  
 **'remindermin'**: reminder\_min})  
 *# adding imput data to the database>>>>>>>>>>>>>>>>>>>>  
 #error handling before data is casted to the database deadeline cant be behind the current date* inte\_value=0  
  
 *#Confirmation for saving the deadline and data verification>>>>>>>>>>>>>>>>>>>>>>>>>>>  
  
  
  
  
  
 #if the daysleft of the dealine are greater than 0 then commit changes to the database* **if** daysleft.days>inte\_value:  
  
 **if** messagebox.askokcancel(**"Saving"**, **"Do you want to save the deadline"**):  
 connection.commit()  
 messagebox.showinfo(**"Success"**, **"Deadline set successfully! Thanks we shall remind you ."**)  
 **else**:  
 print(**"Not saved"**)  
  
  
  
 **else**:  
 messagebox.showerror(**"error"**, **"Data not saved Deadline date cannot be behind the current date "**)  
  
  
  
 *# the code to fetch all information from the data base* row = cursor.fetchall()  
 **for** row **in** cursor.execute(**"select \* from deadline"**):  
 print(row)  
  
 *# closing the database connecion  
  
  
 # showing the dialog /message box after data completion and correctness* **except** ValueError:  
  
  
 messagebox.showerror(**"Error"**, **"Please enter valid details."**)  
 **else**:  
 messagebox.showerror(**"Error"**, **"Please fill all entries. Note:Data must be correct."**)  
*#ALL ERROR HANDLING, VERIFICATION AND DATA ENTRY TO THE DATA BASE IS ACCOMPOLISHED \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
  
# add days left in the data base but remenber the  
# ----------------------  
#SEARCH ENGINE FUNCTION***def** search\_courseWork(event):  
 **try**:  
 search\_input=search\_entry.get()  
 data\_view\_section = css.CTkScrollableFrame(master=coursework\_frame,  
 fg\_color=**"#434344"**,  
 height=290,  
 width=710,  
  
 )  
 data\_view\_section.grid(row=2, column=0, columnspan=2, pady=10, padx=3)  
 *# Entity Labels of the dataview(HEADERS)>>>>>>>>>>>>>>>>>>>>>>>>>>>>  
 # courseName label in dataviewsction* course\_name\_h = css.CTkLabel(master=data\_view\_section,  
 text=**"Course Name"**,  
 fg\_color=**"purple"**,  
 text\_color=**"white"**,  
  
 height=50,  
 width=150  
 )  
 course\_name\_h.grid(row=0, column=0, pady=2, padx=1)  
  
 *# courseName from database:* course = []  
 cursor.execute(**f"select courseName from deadline where courseName like '%{**search\_input**}%' "**)  
 rows = cursor.fetchall()  
 **for** i, row **in** enumerate(rows, start=1):  
 **for** j, value **in** enumerate(row):  
 l = css.CTkLabel(master=data\_view\_section,  
 text=value,  
 fg\_color=**"#017bf5"**,  
 text\_color=**"white"**,  
 height=50,  
 width=150  
 )  
 l.grid(row=i, column=j, pady=2, padx=1)  
 print(value)  
  
 deadline\_label\_h = css.CTkLabel(master=data\_view\_section,  
 text=**"Deadline Date"**,  
 fg\_color=**"purple"**,  
 text\_color=**"white"**,  
 height=50,  
 width=146  
 )  
 deadline\_label\_h.grid(row=0, column=1)  
  
 *# deadline label in dataviewsction* cursor.execute(**f"select Deadline\_date from deadline where courseName like '%{**search\_input**}%'"**)  
 rows = cursor.fetchall()  
 **for** i, row **in** enumerate(rows, start=1):  
 **for** j, value **in** enumerate(row, start=1):  
 l = css.CTkLabel(master=data\_view\_section,  
 text=value,  
 fg\_color=**"#017bf5"**,  
 text\_color=**"white"**,  
 height=50,  
 width=146  
 )  
 l.grid(row=i, column=j, pady=2, padx=1)  
 print(value)  
  
 *# SetDate label in dataviewsection* setDate\_label\_h = css.CTkLabel(master=data\_view\_section,  
 text=**"Set Date"**,  
 fg\_color=**"purple"**,  
 text\_color=**"white"**,  
 height=50,  
 width=146  
 )  
 setDate\_label\_h.grid(row=0, column=2, pady=2, padx=1)  
  
 cursor.execute(**f"select SetDate from deadline where courseName like '%{**search\_input**}%'"**)  
 rows = cursor.fetchall()  
 **for** i, row **in** enumerate(rows, start=1):  
 **for** j, value **in** enumerate(row, start=2):  
 l = css.CTkLabel(master=data\_view\_section,  
 text=value,  
 fg\_color=**"#017bf5"**,  
 text\_color=**"white"**,  
 height=50,  
 width=146  
 )  
 l.grid(row=i, column=j, pady=2, padx=1)  
 print(value)  
  
 *# Reminder time* reminder\_label\_h = css.CTkLabel(master=data\_view\_section,  
 text=**"Remind"**,  
 fg\_color=**"purple"**,  
 text\_color=**"white"**,  
 height=50,  
 width=80  
 )  
 reminder\_label\_h.grid(row=0, column=3, pady=2, padx=1)  
  
 cursor.execute(**f"select remindermin from deadline where courseName like '%{**search\_input**}%'"**)  
 rows = cursor.fetchall()  
 **for** i, row **in** enumerate(rows, start=1):  
 **for** j, value **in** enumerate(row, start=3):  
 l = css.CTkLabel(master=data\_view\_section,  
 text=value,  
 fg\_color=**"#017bf5"**,  
 text\_color=**"white"**,  
 height=50,  
 width=80  
 )  
 l.grid(row=i, column=j, pady=2, padx=1)  
 print(value)  
  
 *# DAYS left label in the dataview* daysleft\_h = css.CTkLabel(master=data\_view\_section,  
 text=**"Days left"**,  
 fg\_color=**"purple"**,  
 text\_color=**"white"**,  
 height=50,  
 width=100  
 )  
 daysleft\_h.grid(row=0, column=4, pady=2, padx=1)  
  
 cursor.execute(**f"select Deadline\_date from deadline where courseName like '%{**search\_input**}%' "**)  
 rows = cursor.fetchall()  
  
 *# accessing the datetime module* x = datetime.datetime.now()  
  
 current\_year = int(x.strftime(**"%Y"**))  
 current\_month = int(x.strftime(**"%m"**))  
 current\_day = int(x.strftime(**"%d"**))  
  
 colour = **"#121312"  
 for** i, row **in** enumerate(rows, start=1):  
 **for** j, value **in** enumerate(row, start=4):  
 current\_date = str(current\_year) + **"/"** + str(current\_month) + **"/"** + str(current\_day)  
 print(value)  
  
 *# coverting to date format>>>>>>>>>>>>>>>>>>>>>>* deadl\_fin = x.strptime(value, **"%Y/%m/%d"**)  
  
 current\_datefin = x.strptime(current\_date, **"%Y/%m/%d"**)  
  
 *# getting days left after subtracting the current date from the deadline* daysleft = deadl\_fin - current\_datefin  
 cl = daysleft.days  
 **if** cl > 3:  
 colour = light  
 **elif** cl >= 1 **and** cl <= 3:  
 colour = **"#9c4141"  
 elif** cl < 1 **and** cl <= -1:  
 colour = **"red"** cl = str(daysleft.days) + **" past"  
  
 else**:  
 print(**"hello"**)  
  
 l = css.CTkLabel(master=data\_view\_section,  
 text=cl,  
 fg\_color=colour,  
 text\_color=**"white"**,  
 height=50,  
 width=100  
 )  
 l.grid(row=i, column=j, pady=2, padx=1)  
 **except** ValueError:  
 messagebox.showerror(**"Not found!"**,**"coursework not found try again"**)  
 print(**"not found"**)  
*# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
# The button command that will refresh the data view section after new data entry  
#-----------------------------------------------------------------------------------------------------------------------***def** refresh\_data\_view\_section\_coursework():  
 data\_view\_section = css.CTkScrollableFrame(master=coursework\_frame,  
 fg\_color=**"#434344"**,  
 height=290,  
 width=710,  
  
  
  
 )  
 data\_view\_section.grid(row=2, column=0, columnspan=2, pady=10, padx=3)  
 *# Entity Labels of the dataview(HEADERS)>>>>>>>>>>>>>>>>>>>>>>>>>>>>  
 # courseName label in dataviewsction* course\_name\_h = css.CTkLabel(master=data\_view\_section,  
 text=**"Course Name"**,  
 fg\_color=**"purple"**,  
 text\_color=**"white"**,  
  
 height=50,  
 width=150  
 )  
 course\_name\_h.grid(row=0, column=0, pady=2, padx=1)  
  
 *# courseName from database:* course = []  
 cursor.execute(**"select courseName from deadline"**)  
 rows = cursor.fetchall()  
 **for** i, row **in** enumerate(rows, start=1):  
 **for** j, value **in** enumerate(row):  
 l = css.CTkLabel(master=data\_view\_section,  
 text=value,  
 fg\_color=**"#017bf5"**,  
 text\_color=**"white"**,  
 height=50,  
 width=150  
 )  
 l.grid(row=i, column=j, pady=2, padx=1)  
 print(value)  
  
 deadline\_label\_h = css.CTkLabel(master=data\_view\_section,  
 text=**"Deadline Date"**,  
 fg\_color=**"purple"**,  
 text\_color=**"white"**,  
 height=50,  
 width=146  
 )  
 deadline\_label\_h.grid(row=0, column=1)  
  
 *# deadline label in dataviewsction* cursor.execute(**"select Deadline\_date from deadline"**)  
 rows = cursor.fetchall()  
 **for** i, row **in** enumerate(rows, start=1):  
 **for** j, value **in** enumerate(row, start=1):  
 l = css.CTkLabel(master=data\_view\_section,  
 text=value,  
 fg\_color=**"#017bf5"**,  
 text\_color=**"white"**,  
 height=50,  
 width=146  
 )  
 l.grid(row=i, column=j, pady=2, padx=1)  
 print(value)  
  
 *# SetDate label in dataviewsection* setDate\_label\_h = css.CTkLabel(master=data\_view\_section,  
 text=**"Set Date"**,  
 fg\_color=**"purple"**,  
 text\_color=**"white"**,  
 height=50,  
 width=146  
 )  
 setDate\_label\_h.grid(row=0, column=2, pady=2, padx=1)  
  
 cursor.execute(**"select SetDate from deadline"**)  
 rows = cursor.fetchall()  
 **for** i, row **in** enumerate(rows, start=1):  
 **for** j, value **in** enumerate(row, start=2):  
 l = css.CTkLabel(master=data\_view\_section,  
 text=value,  
 fg\_color=**"#017bf5"**,  
 text\_color=**"white"**,  
 height=50,  
 width=146  
 )  
 l.grid(row=i, column=j, pady=2, padx=1)  
 print(value)  
  
 *# Reminder time* reminder\_label\_h = css.CTkLabel(master=data\_view\_section,  
 text=**"Remind"**,  
 fg\_color=**"purple"**,  
 text\_color=**"white"**,  
 height=50,  
 width=80  
 )  
 reminder\_label\_h.grid(row=0, column=3, pady=2, padx=1)  
  
 cursor.execute(**"select remindermin from deadline"**)  
 rows = cursor.fetchall()  
 **for** i, row **in** enumerate(rows, start=1):  
 **for** j, value **in** enumerate(row, start=3):  
 l = css.CTkLabel(master=data\_view\_section,  
 text=value,  
 fg\_color=**"#017bf5"**,  
 text\_color=**"white"**,  
 height=50,  
 width=80  
 )  
 l.grid(row=i, column=j, pady=2, padx=1)  
 print(value)  
  
 *# DAYS left label in the dataview* daysleft\_h = css.CTkLabel(master=data\_view\_section,  
 text=**"Days left"**,  
 fg\_color=**"purple"**,  
 text\_color=**"white"**,  
 height=50,  
 width=100  
 )  
 daysleft\_h.grid(row=0, column=4, pady=2, padx=1)  
  
 cursor.execute(**"select Deadline\_date from deadline"**)  
 rows = cursor.fetchall()  
  
 *# accessing the datetime module* x = datetime.datetime.now()  
  
 current\_year = int(x.strftime(**"%Y"**))  
 current\_month = int(x.strftime(**"%m"**))  
 current\_day = int(x.strftime(**"%d"**))  
  
 colour = **"#121312"  
 for** i, row **in** enumerate(rows, start=1):  
 **for** j, value **in** enumerate(row, start=4):  
 current\_date = str(current\_year) + **"/"** + str(current\_month) + **"/"** + str(current\_day)  
 print(value)  
  
 *# coverting to date format>>>>>>>>>>>>>>>>>>>>>>* deadl\_fin = x.strptime(value, **"%Y/%m/%d"**)  
  
 current\_datefin = x.strptime(current\_date, **"%Y/%m/%d"**)  
  
 *# getting days left after subtracting the current date from the deadline* daysleft = deadl\_fin - current\_datefin  
 cl = daysleft.days  
 **if** cl > 3:  
 colour = light  
 **elif** cl >= 1 **and** cl <= 3:  
 colour = **"#9c4141"  
 elif** cl < 1 **and** cl <= -1:  
 colour = **"red"** cl = str(daysleft.days) + **" past"  
  
 else**:  
 print(**"hello"**)  
  
 l = css.CTkLabel(master=data\_view\_section,  
 text=cl,  
 fg\_color=colour,  
 text\_color=**"white"**,  
 height=50,  
 width=100  
 )  
 l.grid(row=i, column=j, pady=2, padx=1)  
  
 *# Ellapsed label in the  
  
  
  
  
#--------------------------------------------------------------------------------------------------------------------  
  
  
  
#A Function that clears all data in entries to null***def** clear\_entries():  
  
 courseUnitInput.delete(0,15)  
  
 deadline\_yearInput.delete(0,15)  
  
  
  
 deadline\_monthInput.delete(0,15)  
  
  
  
 deadline\_dayInput.delete(0,15)  
  
  
  
 reminder\_minute\_Input.delete(0,15)  
  
*#-----------------------------------------------------------------------------------------------------------------------  
  
# swapping to options of shutting down the computer***def** power():  
 coursework\_frame.grid\_forget()  
 display\_frame.grid\_forget()  
 others\_frame.grid()  
 others\_frame.grid\_propagate(**False**)  
 power\_btn.configure(fg\_color=**"#434344"**)  
 display\_btn.configure(fg\_color=**"#017bf5"**)  
 about\_us\_btn.configure(fg\_color=**"#017bf5"**)  
 coursework\_btn.configure(fg\_color=**"#017bf5"**)  
  
  
*#-----------------------------------------------------------------------------------------------------------------------***def** shutdown():  
 **return** os.system(**"shutdown /s /hybrid"**)  
  
  
**def** restart():  
 **return** os.system(**"shutdown /r /t 1"**)  
  
  
**def** hibernate():  
 *# return os.system("shutdown /h")* **return** os.system(**"dxdiag"**)  
  
*#-----------------------------------------------------------------------------------------------------------------------  
# NOTIFICATION  
# Notify the user about the deadline of any course work  
# def notify():  
# notification.notify(  
# title=" Networking course work ",  
# message=f"""  
# daysleft : {daysleft} Hon Deus  
#  
#  
# """  
# )  
# win.after(remind\_min, notify)  
#  
  
# FUNCTION-----------------------------------------------------*win = css.CTk()  
win.geometry(**"900x500"**)  
win.title(**"Eyes Ug"**)  
win.resizable(**False**, **False**)  
*# icons and images  
  
  
# set Left side MASTER frame with options\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**side\_frame = css.CTkFrame(master=win, width=160, height=500, fg\_color=**"white"**, corner\_radius=0)  
side\_frame.configure(width=160, height=500)  
side\_frame.grid(row=0, column=0)  
side\_frame.columnconfigure(0, weight=1)  
side\_frame.rowconfigure(0, weight=1)  
*# side\_frame.grid\_propagate(False)  
# widgets on the Left side frame  
# display button*display\_btn = css.CTkButton(master=side\_frame, text=**"Display"**, corner\_radius=0, width=160, height=120,  
 fg\_color=**"#434344"**,  
 text\_color=**"white"**,  
 hover\_color=**"#272727"**,  
 *# image=icon* command=display,  
 hover=**False** )  
  
*# Course Work*coursework\_btn = css.CTkButton(master=side\_frame, text=**"Coursework"**, width=160, height=120,  
  
 text\_color=**"white"**,  
 hover=**False**,  
 corner\_radius=0,  
 command=coursework  
  
 )  
  
*# Power button*power\_btn = css.CTkButton(master=side\_frame, text=**"Power Options"**, width=160, height=120,  
 hover=**False**,  
 corner\_radius=0,  
 command=power  
 )  
  
*# About us button*about\_us\_btn = css.CTkButton(master=side\_frame, text=**"About us"**, width=160, height=141,  
 hover=**False**,  
 corner\_radius=0)  
  
*# END OF LEFT SIDE FRAME\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
  
# grid geometry for options of left side frame*display\_btn.grid(row=0, column=0)  
coursework\_btn.grid(row=1, column=0)  
power\_btn.grid(row=2, column=0)  
about\_us\_btn.grid(row=3, column=0)  
  
*# SET Right side MASTER frame with options\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**right\_frame = css.CTkFrame(master=win, width=900, height=500, corner\_radius=0)  
  
*# DISPLAY FRAME--------------------------------------------------DISPLAY FRAME*display\_frame = css.CTkFrame(master=right\_frame, width=800, height=500, fg\_color=**"#434344"**, corner\_radius=0)  
  
*# brightness label and slider>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>*bright\_label = css.CTkLabel(master=display\_frame, text=**"Brightness"**, width=100, height=100, corner\_radius=0,  
 text\_color=**"white"**,  
 font=(**'Arial'**, 20)  
 )  
bright\_slider = css.CTkSlider(master=display\_frame, height=20, from\_=0, to=100, width=400,  
 hover=**False**,  
 command=change\_brightness,  
  
 )  
  
*# sharpness label and slider>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>*sharpness\_label = css.CTkLabel(master=display\_frame, text=**"Sharpness"**, width=100, height=100, corner\_radius=0,  
 text\_color=**"white"**,  
 font=(**"Apple"**, 20)  
 )  
sharpness\_slider = css.CTkSlider(master=display\_frame, height=20, from\_=0, to=100, width=400,  
 *# command=change\_sharpness* )  
  
*# Developer Name:>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>*dev\_name\_label = css.CTkLabel(master=display\_frame, text=**"Developer : MUKIIBI DEUS "**,  
 font=(**"Arial"**, 20),  
  
 text\_color=**"#fff"**,  
 height=20)  
  
*# developer Contact>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>*dev\_no\_label = css.CTkLabel(master=display\_frame, text=**"Contact:+256702917121"**, height=20,  
 text\_color=**"#fff"** )  
  
*# developer University*dev\_university\_label = css.CTkLabel(master=display\_frame, text=**"University:Uganda Martyrs"**,  
 text\_color=**"#fff"**,  
 height=20)  
  
*# COURSE WORK FRAME AND OPTIONS--------------------------------------------*coursework\_frame = css.CTkFrame(master=right\_frame,  
 fg\_color=light,  
 width=900,  
 height=500,  
 corner\_radius=**False** )  
  
*# PAD ONE IN COURSE WORK >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>*cousre\_pad1 = css.CTkFrame(master=coursework\_frame,  
 width=204,  
 height=180,  
 corner\_radius=10,  
 fg\_color=**"#434344"** )  
cousre\_pad1.grid(row=0, column=0, padx=20)  
cousre\_pad1.grid\_propagate(**False**)  
*# Pad1 Title for Course work>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>*cWorkTitle1 = css.CTkLabel(master=cousre\_pad1,  
 text=**"Set deadline"**,  
 width=204,  
 height=30,  
 corner\_radius=1,  
 fg\_color=**"#017bf5"**,  
 text\_color=**"white"** )  
cWorkTitle1.grid(row=0, column=0, columnspan=4)  
*# COURSE UNIT NAME*course\_unit\_label = css.CTkLabel(master=cousre\_pad1,  
 fg\_color=**"white"**,  
 width=60,  
 text=**"Course"** )  
course\_unit\_label.grid(row=1, column=0, pady=6)  
*# reminder input>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>*courseUnitInput = css.CTkEntry(master=cousre\_pad1, fg\_color=**"white"**,  
 text\_color=**"black"**,  
 width=142,  
 placeholder\_text=**"course Name"**,  
 height=30  
 )  
courseUnitInput.grid(row=1, column=1, padx=1, columnspan=3)  
*# pad deadline label for deadline inputs>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>*deadline\_label = css.CTkLabel(master=cousre\_pad1,  
 width=60,  
 fg\_color=**"white"**,  
 text=**"Deadline"**,  
 )  
deadline\_label.grid(row=2, column=0)  
  
*# DEAD LINE INPUTS ONE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*SUBHEADING  
# deadline year input>>>>>>>>>>>>>>>*deadline\_yearInput = css.CTkEntry(master=cousre\_pad1, fg\_color=**"white"**,  
 text\_color=**"black"**,  
 placeholder\_text=**"year"**,  
 width=50,  
 height=30  
 )  
deadline\_yearInput.grid(row=2, column=1)  
*# deadline month input>>>>>>>>>>>>>>>*deadline\_monthInput = css.CTkEntry(master=cousre\_pad1, fg\_color=**"white"**,  
 text\_color=**"black"**,  
 placeholder\_text=**"month"**,  
 width=49,  
 height=30  
 )  
deadline\_monthInput.grid(row=2, column=2)  
*# deadline month input>>>>>>>>>>>>>>>*deadline\_dayInput = css.CTkEntry(master=cousre\_pad1, fg\_color=**"white"**,  
 text\_color=**"black"**,  
 placeholder\_text=**"day"**,  
 width=40,  
 height=30  
 )  
deadline\_dayInput.grid(row=2, column=3)  
*# DEALINE REMINDER\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*SUBHEADING*reminder\_label = css.CTkLabel(master=cousre\_pad1,  
 fg\_color=**"white"**,  
 width=60,  
 text=**"Reminder"** )  
reminder\_label.grid(row=3, column=0)  
*# reminder input>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>*reminder\_minute\_Input = css.CTkEntry(master=cousre\_pad1, fg\_color=**"white"**,  
 text\_color=**"black"**,  
 width=142,  
 placeholder\_text=**"minutes"**,  
 height=30  
 )  
reminder\_minute\_Input.grid(row=3, column=1, pady=10, padx=1, columnspan=3)  
*# SAVE BUTTON>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>*save\_coursework\_changes = css.CTkButton(master=cousre\_pad1,  
 text=**"Save"**,  
 width=50,  
 fg\_color=**"#017bf5"**,  
 command=getDeadlineDetailsPad1  
  
 )  
save\_coursework\_changes.grid(row=4, column=1)  
*# CANCEL BUTTON>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>*exit\_btn = css.CTkButton(master=cousre\_pad1,  
 text=**"clear"**,  
 width=87,  
 fg\_color=**"#017bf5"**,  
 command=clear\_entries  
  
 )  
exit\_btn.grid(row=4, column=2, columnspan=2)  
*# CANCEL BUTTON>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>  
#COURSE WORKPAD DONE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
  
#SEARCH FRAME*search\_frame=css.CTkFrame(master=coursework\_frame,  
 width=300,  
 height=100,  
 fg\_color=**"gray"** )  
search\_frame.grid(row=0,column=1)  
search\_frame.grid\_propagate(**False**)  
  
*#search Entry>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>*search\_entry=css.CTkEntry(master=search\_frame,  
 placeholder\_text=**"search course work"**,  
 height=30,  
 width=199,  
  
 )  
search\_entry.grid(row=0,column=0,pady=10)  
*#search Entry button in search frame*search\_button=css.CTkButton(master=search\_frame,  
 height=30,  
 width=80,  
 text=**"search"**,  
 command=search\_courseWork  
  
 )  
search\_button.grid(row=0,column=1,pady=10,padx=5)  
search\_entry.bind(**"<Key>"**,search\_courseWork)  
  
*#Refresh button in coursework frame to repick data from the datbase implement new changes if present*refresh\_data\_view\_section\_coursework=css.CTkButton(master=search\_frame,  
 text=**"Refresh"**,  
 command=refresh\_data\_view\_section\_coursework,  
 hover\_color=**"#434344"**,  
 fg\_color=**"purple"**,  
 width=80,  
 height=30  
  
 )  
refresh\_data\_view\_section\_coursework.grid(row=1,column=1,pady=20)  
  
  
*# DATA VIEW FROM THE DATABASE\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*data\_view\_section=css.CTkScrollableFrame(master=coursework\_frame,  
 fg\_color=**"#434344"**,  
 height=290,  
 width=710  
  
  
  
 )  
data\_view\_section.grid(row=2,column=0,columnspan=2,pady=10,padx=3)  
*#Entity Labels of the dataview(HEADERS)>>>>>>>>>>>>>>>>>>>>>>>>>>>>  
#courseName label in dataviewsction*course\_name\_h=css.CTkLabel(master=data\_view\_section,  
 text=**"Course Name"**,  
 fg\_color=**"purple"**,  
 text\_color=**"white"**,  
  
 height=50,  
 width=120  
 )  
course\_name\_h.grid(row=0,column=0,pady=2,padx=1)  
  
  
*#courseName from database:*course=[]  
cursor.execute(**"select courseName from deadline"**)  
rows=cursor.fetchall()  
**for** i,row **in** enumerate(rows,start=1):  
 **for** j,value **in** enumerate(row):  
  
 l = css.CTkLabel(master=data\_view\_section,  
 text=value,  
 fg\_color=**"#017bf5"**,  
 text\_color=**"white"**,  
 height=50,  
 width=120  
 )  
 l.grid(row=i, column=j, pady=2,padx=1)  
 print(value)  
  
  
  
  
deadline\_label\_h=css.CTkLabel(master=data\_view\_section,  
 text=**"Deadline Date"**,  
 fg\_color=**"purple"**,  
 text\_color=**"white"**,  
 height=50,  
 width=146  
 )  
deadline\_label\_h.grid(row=0,column=1)  
  
  
*#deadline label in dataviewsction*cursor.execute(**"select Deadline\_date from deadline"**)  
rows=cursor.fetchall()  
**for** i,row **in** enumerate(rows,start=1):  
 **for** j,value **in** enumerate(row,start=1):  
  
 l = css.CTkLabel(master=data\_view\_section,  
 text=value,  
 fg\_color=**"#017bf5"**,  
 text\_color=**"white"**,  
 height=50,  
 width=146  
 )  
 l.grid(row=i, column=j, pady=2,padx=1)  
 print(value)  
  
*#SetDate label in dataviewsection*setDate\_label\_h=css.CTkLabel(master=data\_view\_section,  
 text=**"Set Date"**,  
 fg\_color=**"purple"**,  
 text\_color=**"white"**,  
 height=50,  
 width=146  
 )  
setDate\_label\_h.grid(row=0,column=2,pady=2,padx=1)  
  
  
cursor.execute(**"select SetDate from deadline"**)  
rows=cursor.fetchall()  
**for** i,row **in** enumerate(rows,start=1):  
 **for** j,value **in** enumerate(row,start=2):  
  
 l = css.CTkLabel(master=data\_view\_section,  
 text=value,  
 fg\_color=**"#017bf5"**,  
 text\_color=**"white"**,  
 height=50,  
 width=146  
 )  
 l.grid(row=i, column=j, pady=2,padx=1)  
 print(value)  
  
  
*#Reminder time*reminder\_label\_h=css.CTkLabel(master=data\_view\_section,  
 text=**"Remind"**,  
 fg\_color=**"purple"**,  
 text\_color=**"white"**,  
 height=50,  
 width=80  
 )  
reminder\_label\_h.grid(row=0,column=3,pady=2,padx=1)  
  
  
cursor.execute(**"select remindermin from deadline"**)  
rows=cursor.fetchall()  
**for** i,row **in** enumerate(rows,start=1):  
 **for** j,value **in** enumerate(row,start=3):  
  
 l = css.CTkLabel(master=data\_view\_section,  
 text=value,  
 fg\_color=**"#017bf5"**,  
 text\_color=**"white"**,  
 height=50,  
 width=80  
 )  
 l.grid(row=i, column=j, pady=2,padx=1)  
 print(value)  
  
  
  
  
  
*#DAYS left label in the dataview*daysleft\_h=css.CTkLabel(master=data\_view\_section,  
 text=**"Days left"**,  
 fg\_color=**"purple"**,  
 text\_color=**"white"**,  
 height=50,  
 width=100  
 )  
daysleft\_h.grid(row=0,column=4,pady=2,padx=1)  
  
  
cursor.execute(**"select Deadline\_date from deadline"**)  
rows=cursor.fetchall()  
  
*# accessing the datetime module*x = datetime.datetime.now()  
  
current\_year = int(x.strftime(**"%Y"**))  
current\_month = int(x.strftime(**"%m"**))  
current\_day = int(x.strftime(**"%d"**))  
  
  
  
colour=**"#121312"  
for** i,row **in** enumerate(rows,start=1):  
 **for** j,value **in** enumerate(row,start=4):  
 current\_date = str(current\_year) + **"/"** + str(current\_month) + **"/"** + str(current\_day)  
 print(value)  
  
 *# coverting to date format>>>>>>>>>>>>>>>>>>>>>>* deadl\_fin = x.strptime(value,**"%Y/%m/%d"**)  
  
 current\_datefin = x.strptime(current\_date, **"%Y/%m/%d"**)  
  
 *# getting days left after subtracting the current date from the deadline* daysleft = deadl\_fin - current\_datefin  
 cl = daysleft.days  
 **if** cl > 3:  
 colour = light  
 **elif** cl >= 1 **and** cl<=3:  
 colour = **"#9c4141"  
 elif** cl<1 **and** cl<=-1:  
 colour = **"red"** cl=str(daysleft.days)+**" past"  
  
 else**:  
 print(**"hello"**)  
  
  
  
 l = css.CTkLabel(master=data\_view\_section,  
 text=cl,  
 fg\_color=colour,  
 text\_color=**"white"**,  
 height=50,  
 width=100  
 )  
 l.grid(row=i, column=j, pady=2,padx=1)  
  
  
*#Ellapsed label in the  
  
#END OF THE COURSE WORK FRAME  
#-----------------------------------------------------------------------------------------------------------------------  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
# Grid geometry for right side frame with options  
  
  
# OTHERS BTN FRAME-------------------------------------------------------------------------------*others\_frame = css.CTkFrame(master=right\_frame,  
 width=700,  
 height=500,  
 fg\_color=**"#191c1f"**,  
 corner\_radius=**False**)  
  
*# others frame options*shutdwon\_btn = css.CTkButton(master=others\_frame, height=100, width=100, text=**"Shutdown"**,  
 fg\_color=**"#ffd901"**,  
 text\_color=**"black"**,  
 hover=**False**,  
  
 command=shutdown  
  
 )  
restart\_btn = css.CTkButton(master=others\_frame, height=100, width=100, text=**"Restart"**,  
 fg\_color=**"#036630"**, hover=**False**,  
 command=restart  
 )  
hibernate\_btn = css.CTkButton(master=others\_frame, height=100, width=100, text=**"Hibernate"**,  
 fg\_color=**"#032e66"**,  
 hover=**False**,  
 command=hibernate  
 )  
  
*# grid geometry options for the right side frame  
# ---------------------------------------*right\_frame.grid(row=0, column=1)  
right\_frame.grid\_propagate(**False**)  
*# ---------------------------------------  
# ---------------------------------------*display\_frame.grid()  
display\_frame.grid\_propagate(**False**)  
*# ----------------------------------------*bright\_label.grid(row=0, column=0)  
bright\_slider.grid(row=0, column=1)  
sharpness\_label.grid(row=1, column=0)  
sharpness\_slider.grid(row=1, column=1)  
dev\_name\_label.grid(row=2, column=0, columnspan=3, sticky=**"W"**, padx=40, pady=20)  
dev\_no\_label.grid(row=3, column=0, columnspan=3, sticky=**"W"**, padx=40, pady=20)  
dev\_university\_label.grid(row=4, column=0, columnspan=3, sticky=**"W"**, padx=40, pady=20)  
*# COURSE WORK FRAME AND OPTIONS--------------------------------------------*coursework\_frame.grid()  
*# coursework\_frame.grid\_propagate(False)  
# -----------------------------------------*others\_frame.grid()  
others\_frame.grid\_propagate(**False**)  
*# -----------------------------------------*shutdwon\_btn.grid(row=1, column=0, padx=30, pady=160)  
restart\_btn.grid(row=1, column=1, padx=20, pady=160)  
hibernate\_btn.grid(row=1, column=2, padx=20, pady=160)  
  
*# NOTIFY FUNCTIONS  
# notify()*win.mainloop()  
connection.close()