الأكاديمية العربية للعلوم والتكنولوجيا والنقل البحري

Arab Academy for Science, Technology & Maritime Transport

Report on 12th project SPL (python)

Name: Elza Morgan, Youssef El Tabib, Mohamed Abozaid, Mohamed Yasser

Registration No.: 18100260, 18102698, 18102968, 18102456

Course code: CS445

Department: CS

Dr: Yasser Fouad

TA: Mahmoud El Morshedy

Contents

Idea of our project: 3

Explanation of our code: 3

Images of our Application Running.....15

.....15

Idea of our project:

Our idea for this project is creating a system for university that can register, view, delete and update students by using python language. When it comes to registering a student, you can enter their information manually or by autofill. This autofill is a function that we have created that reads from any json file type and uploads this specifically file of that student automatically. Moreover, all of this information about the students will be added to the database by using MySQL database. The interaction with the system will be through GUI.

Explanation of our code:

```
20  def MainScreen():
21      #used to create the screen which is the big window
22      #down here we have some properties or attributes for creating the screen
23      screen = Tk()
24      screen.title('Registration System')
25      screen.geometry('600x300')
26      screen.config(bg="#447c84")
27      #-----------------------------------
28
29      #that's used to create the fram which is found inside the screen, it's like a small window
30      #some properties or attributes are used for screating the frame
31      frame = Frame(screen, height=400,width=300,padx=20, pady=20)
32      frame.config(bg="white")
33      frame.pack_propagate(False)
34      frame.pack(expand=True)
35
36      #Label us used to create like a text
37      Label(
38          frame, #here it is used to specify where the label will be found
39          #some attributes or properties added for the label
40          text="Registration System",
41          font=("Times", "24", "bold")
42      ).grid(row=0,  columnspan=6, pady=10) #used to specify the postion of the label
43
44      #creating a button for that label
45      Enter = Button(
46          frame,
47          text="Enter System",
48          padx=20, pady=10,
49          relief=RAISED, #used to make it like bevel
49          relief=RAISED, #used to make it like bevel
50          font=("Times", "16", "bold"),
51          #this line means when I click on this button destroy the screen it is at and then go to function EnterSystem
52          command=lambda:[screen.destroy(),EnterSystem()]
53          ).grid(row=2, column=2, pady=10)
54      #used to make the screen full screen
55      screen.state("zoomed")
56      screen.mainloop() #out of every screen created we must close it or end it in order to move onto the next screen
57
```

Main() function

This is Main Function that will be displayed, when you run the code. Inside we have created screen to create the layout of the window, we also gave it some properties or attributes to look representable. Inside the screen we have created a frame which is the small box inside the screen we gave it some attributes and properties and we set the position for it. Inside the frame we have created label which is like creating a text we have also added some properties or attributes for it. After we have created a button where it is used to destroy or closes the current screen and calls the next function which is EnterSystem() and that will be at line 52. The function lambda means is that the function will occur when variable or method command is called which is by pressing onto the button. At line 55 means make the screen/window bigger or Fullscreen and at line 56 means close the screen of main menu and enter the new screen called from line 52.

```
58      #thats the function EnterSystem that will be called from line 52
59  def EnterSystem():
60      #created another screen
61      screen = Tk()
62      screen.title('Registration System')
63      screen.geometry('400x500')
64      screen.config(bg="#447c84")
65      #created another frame
66      frame = Frame(screen, height=400,width=300,padx=20, pady=20)
67      frame.config(bg="white")
68      frame.pack_propagate(False)
69      frame.pack(expand=True)
70
71      #a text that is found inside the frame
72      Label(
73          frame,
74          text="Registration System",
75          font=("Times", "24", "bold")
76      ).grid(row=0,  columnspan=6, pady=5)
77
78      #button for the registeration which is found isnide the frame as well
79      Reg = Button(
80          frame,
81          width=15,
82          text="Register a Student",
83          padx=20, pady=5,
84          relief=RAISED,
85          font=("Times", "16", "bold"),
86          #this line means when I click on this button destroy the screen it is at and then go to function Registration
87          command=lambda :[screen.destroy(),Registration()]
```

```
87          command=lambda :[screen.destroy(),Registration()]
88          ).grid(row=1, column=1, pady=15)
89
90          #another button created for displaying the student info
91      Reg = Button(
92          frame, width=15,
93          text="Retrieve Student Info",
94          padx=20, pady=10,
95          relief=RAISED,
96          font=("Times", "16", "bold"),
97          #this line means when I click on this button destroy the screen it is at and then go to function readFromjson files
98          command=lambda:[screen.destroy(), readFromJSON()]
99          ).grid(row=2, column=1, pady=15)
100
101         #button that is used to update the student if needed
102     Upd = Button(
103         frame,
104         width=15,
105         text="Update Student Info",
106         padx=20, pady=10,
107         relief=RAISED,
108         font=("Times", "16", "bold"),
109         #this line means when I click on this button destroy the screen it is at and then go to function updateStudent
110         command=lambda:[screen.destroy(), updateStudent()]
111         ).grid(row=3, column=1, pady=15)
112
113     #button that is used to delete any student if needed
114     delete = Button(
115         frame,
116         width=15,
117         text="Delete Student",
```

```
117         text="Delete Student",
118         padx=20, pady=10,
119         relief=RAISED,
120         font=("Times", "16", "bold"),
121         #this line means when I click on this button destroy the screen it is at and then go to function deleteStudent
122         command=lambda:[screen.destroy(),deleteStudent()]
123         ).grid(row=4, column=1, pady=15)
124
125
126     screen.state("zoomed")
127     screen.mainloop()
128     #screen will be closed since we are done with this specifc screen
```

EnterSystem() Function

We have created another screen for it, inside it we have created a frame. Inside the frame we have created a label as text which is “registration system”. We have created several buttons in that frame. First for registering student, this function registration will be called at line 87. Second button to retrieve student information by reading from json file, this function readFromJson will be called at line 98. Third button is created is for updating any information about this specific student, this function updateStudent will be called at line 110. Last button created used to delete any student from the database when needed, this function deleteStudent will be called at line 122. At line 126 means make the

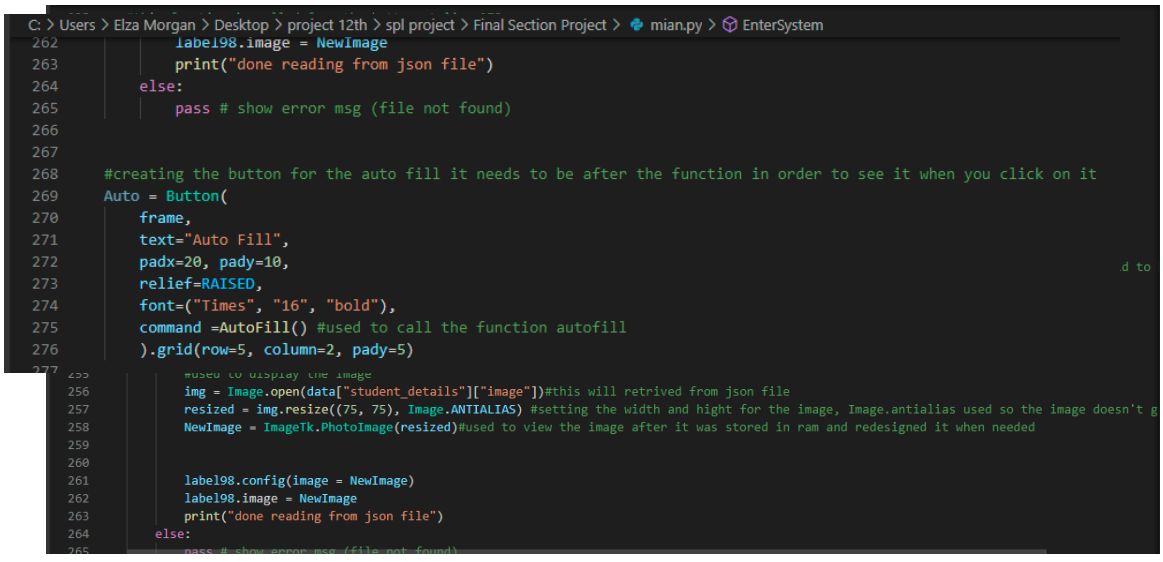
screen/window bigger or Fullscreen and at line 127 means close the screen of main menu and enter the new screen.

```
#function for registration that will be called from line 87
def Registration() :
    #another screen created for this function
    screen = Tk()
    screen.title('Registration System')
    screen.geometry('600x500')
    screen.config(bg="#447c84")

    #image of back button
    img = Image.open("C:/Users/Elza Morgan/Desktop/project 12th/spl project/Untitled-4.png") #file path for the image
    resized = img.resize((50, 50), Image.ANTIALIAS) #setting the width and hight for the image, Image.antialias used so the image doesn't get pix
    newimg = ImageTk.PhotoImage(resized) #used to view the image after it was stored in ram and redesigned it when needed

    #funciton for the back button when clicked on
    back = Button(
        screen,
        image=newimg,
        padx=20, pady=10,
        relief=RAISED,
        borderwidth=0,
        background="#447c84",
       # this line means when I click on this button destroy the screen it is at and then go back to EnterSystem which is the main menu
        command=lambda:[screen.destroy(),EnterSystem()]
        )
    back.place(x=30,y=40)

    frame = Frame(screen, height=400,width=10,padx=20, pady=20)
    frame.config(bg="white")
    frame.pack_propagate(False)
    frame.pack(expand=True)
    photo = Image.open("C:/Users/Elza Morgan/Desktop/project 12th/spl project/avatar(1).jpg")#used to display the icon of profile in regsiter fu
    resized = photo. resize((110, 75), Image.ANTIALIAS) #setting the width and hight for the image, Image.antialias used so the image doesn't get
    NewImage = ImageTk.PhotoImage(resized)#used to view the image after it was stored in ram and redesigned it when needed

    label98 = Label(screen,bg="white",image=NewImage,width=70,height=70)
    label98.pack()
    label98.place(bordermode=OUTSIDE, x=820,y=250) # label for the image

    #used to create the label as a title
    Label(
    frame,
    text="Register a student",
    font=("Times", "24", "bold")
    ).grid(row=0, columnspan=3, pady=10)

    #used to create the label for button student id
    Label(
    frame,
    text='Student Id',
    font=("Times", "14")
    ).grid(row=1, column=0, pady=5)

    #this line we have created a global varibable in order to use it if needed
    global id1;id1= Entry(frame, width=30) #this line is used to create a textfiled
    id1.pack()
    id1.grid(row=1,column=1)#setting the postion for the textfield box

    #creating label or a text for in First Name
    Label(
    frame,
    text='First Name',
    font=("Times", "14")
    ).grid(row=2, column=0, pady=5)

    #this line we have created a global varibable in order to use it if needed
    global fname;fname = Entry(frame, width=30)#this line is used to create a textfiled
    fname.pack()
    fname.grid(row=2, column=1)#setting the postion for the textfield box

    #creating label or a text for in Last Name
    Label(
    frame,
    text='Last Name',
    font=("Times", "14")
    ).grid(row=3, column=0, pady=5)

    #this line we have created a global varibable in order to use it if needed
    global lname;lname = Entry(frame, width=30)#this line is used to create a textfiled
    lname.pack()
    lname.grid(row=3, column=1)#setting the postion for the textfield box

    #creating label or a text for Email
    Label(
    frame,
    text='Email Address',
    font=("Times", "14")
    ).grid(row=4, column=0, pady=5)

    #this line we have created a global varibable in order to use it if needed
    global email; email = Entry(frame, width=30)#this line is used to create a textfiled
    email.pack()
```

```
222     email.pack()
223     email.grid(row=4, column=1)#setting the postion for the textfield box
224
225     #used to create button for register
226     finalReg = Button(
227         frame,
228         text="Register",
229         padx=20, pady=10,
230         relief=RAISED,
231         font=("Times", "16", "bold"),
232         command=addStudent() #this is used to call the addStudent function
233         ).grid(row=5, column=1, pady=5)
```

Registration() function:

We have created another screen for this function using suitable properties or attributes. Then we inserted an image from the folder, image of back button so when you click on it, it returns you to the main menu screen. Then we have used this image and created a button by variable name called “back” and did the calling back button to main menu at line 153 but before that It will destroy the current screen. Then we have created a frame inside the screen, inside we have added an avatar icon on the right corner. At line 163 the image will be stored at ram and we will do all the editing for this image and then at line 164 means it will take this image that was stored and then it will display it onto the screen. We have created several labels inside this frame. First label created for the student Id we set the suitable property or attribute for it then we have created text filed for it and we have set the suitable attribute or property for it. The variable for it will be set to global because we want to access it later on. Second label created is for first name of the student and we have set text field for it as well as well as the variable that is created is set to be global in order to use it later on. Third label is created for Last Name of the student and we have set text field for it as well as well as the variable that is created is set to be global in order to use it later on. Fourth label is created is email of the student and we have set text field for it as well as well as the variable that is created is set to be global in order to use it later on. After entering the information of the student manually there will be a button used to call the function addstudent().



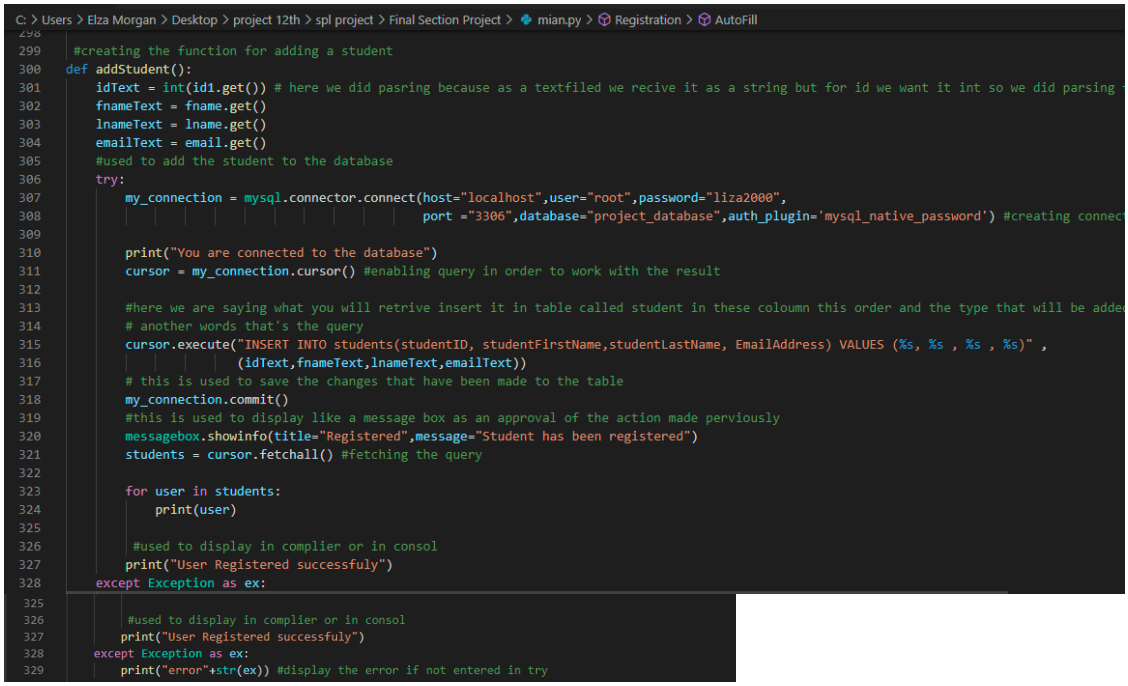
AutoFill() function

First of all we will read from any type of Json file, when the file is not empty we will load the file that was uploaded from json files and stored in another variable called “data” line from 243 to 246 deletes anything found in text field which overrides it when needed, we included 0 to start from the very start. After the json file is read and stored in variable “data” in order to view it we have called the name of the textfield.insert inside it we have mentioned the position to start to view it and have called the variable ”data” and the name of the object or structure created in json file and then we specify the field and insert the data.from line 256 to 258 is used to retrieve the image path added in json file and then store it and resize it and then add upload it to the system and make it appear. After all, if the file is empty, it will enter the in else and display an error. After creating the function autofill and button is created and to call this function is called at line 275.Function autofill comes under function Registration.

Clear() function

Used to call the text field variables that were known to be globally and we have set the position of from where to clear which the number 0. Then we have created a button so that when you click on it will the function clear and that will be at line 292.

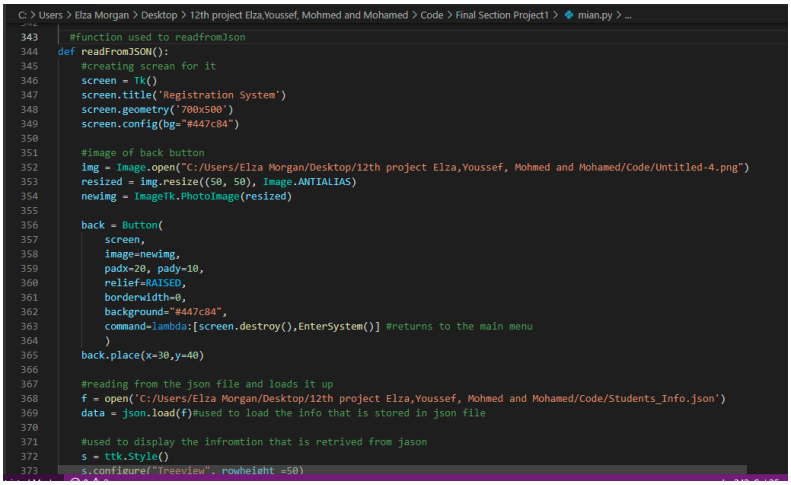
Afterall, at line 295 used to make the screen full screen and 296 used to close the screen for registering student.



```
C: > Users > Elza Morgan > Desktop > project 12th > spl project > Final Section Project > mian.py > Registration > AutoFill
298
299   #creating the function for adding a student
300   def addStudent():
301       idText = int(id1.get()) # here we did pasring because as a textfiled we recive it as a string but for id we want it int so we did parsing
302       fnameText = fname.get()
303       lnameText = lname.get()
304       emailText = email.get()
305       #used to add the student to the database
306       try:
307           my_connection = mysql.connector.connect(host="localhost",user="root",password="liza2000",
308                                                   port ="3306",database="project_database",auth_plugin='mysql_native_password') #creating connect
309
310           print("You are connected to the database")
311           cursor = my_connection.cursor() #enabling query in order to work with the result
312
313           #here we are saying what you will retrive insert it in table called student in these coloumn this order and the type that will be adde
314           # another words that's the query
315           cursor.execute("INSERT INTO students(studentID, studentFirstName,studentLastName, EmailAddress) VALUES (%s, %s , %s , %s)" ,
316                          (idText,fnameText,lnameText,emailText))
317           # this is used to save the changes that have been made to the table
318           my_connection.commit()
319           #this is used to display like a message box as an approval of the action made perviously
320           messagebox.showinfo(title="Registered",message="Student has been registered")
321           students = cursor.fetchall() #fetching the query
322
323           for user in students:
324               print(user)
325
326            #used to display in complier or in consol
327           print("User Registered successfuly")
328       except Exception as ex:
325
326           #used to display in complier or in consol
327           print("User Registered successfuly")
328       except Exception as ex:
329           print("error"+str(ex)) #display the error if not entered in try
```

addStudent() function

we will take all the information that was added in the text field and added to a variable that will be used in this function. For id we have made parsing since we want to use it as integer not as a string. Then it will enter the “try” and at line 307 that’s how we connect to the database and then we tried to open connection to enable querying in the database. After words we have created, a query saying insert to the table student’s information that was retrieved from the text field variable and then save the table. Then we save the table since we inserted some stuff to the table. Message will be displayed that this student is added to the database. Then we will take every row from the database and then added to the variable that will be converted to list which is called “students”. Then we will create a for loop to display all the student’s information. If it didn’t enter the try and entered in the exception then it will print the name of the exception error in the console or the terminal.



```
C: > Users > Elza Morgan > Desktop > 12th project Elza,Youssef, Mohmed and Mohamed > Code > Final Section Project1 > mian.py > ...
343   #function used to readfromJson
344   def readFromJSON():
345       #creating screan for it
346       screen = Tk()
347       screen.title('Registration System')
348       screen.geometry('700x500')
349       screen.config(bg="#447c84")
350
351       #image of back button
352       img = Image.open("C:/Users/Elza Morgan/Desktop/12th project Elza,Youssef, Mohmed and Mohamed/Code/Untitled-4.png")
353       resized = img.resize((50, 50), Image.ANTIALIAS)
354       newimg = ImageTk.PhotoImage(resized)
355
356       back = Button(
357           screen,
358           image=newimg,
359           padx=20, pady=10,
360           relief=RAISED,
361           borderwidth=0,
362           background="#447c84",
363           command=lambda:[screen.destroy(),EnterSystem()] #returns to the main menu
364           )
365       back.place(x=30,y=40)
366
367       #reading from the json file and loads it up
368       f = open('C:/Users/Elza Morgan/Desktop/12th project Elza,Youssef, Mohmed and Mohamed/Code/Students_Info.json')
369       data = json.load(f)#used to load the info that is stored in json file
370
371       #used to display the infromtion that is retrived from jason
372       s = ttk.Style()
373       s.configure("Treeview", rowheight =50)
```

```
C: > Users > Elza Morgan > Desktop > 12th project Elza,Youssef, Mohmed and Mohamed > Code > Final Section Project1 > mian.py > ...
372       s = ttk.Style()
373       s.configure("Treeview", rowheight =50)
374       tree = ttk.Treeview(screen, columns=("First Name","Last Name", "ID", "Email"),height=len(data["student_details"]))
375       #used to specify the order of the coloumn                                    #used to make the table in treeview to be dynamic not fixed
376       tree.heading("#1", text="First Name")
377       tree.heading("#2", text="Last Name")
378       tree.heading("#3", text="ID")
379       tree.heading("#4", text="Email")
380
381       tree.column("#0", stretch='No', width=0)
382       tree.column('First Name',anchor='center', width=200)
383       tree.column('Last Name',anchor='center', width=200)
384       tree.column('ID',anchor='center', width=200)
385       tree.column('Email', anchor='center', width=350)
386       tree.pack(fill=BOTH, expand=True)
387       tree.place(x=230,y=190) #sets overall table in the screen
388
389       #used to put the data inside the table
390       for i in data["student_details"]:
391           tree.insert("", "end" ,values=(i["student_fname"],i["student_lname"], i["student_id"], i["student_email"]))
392       f.close()#closes file that was open in line 353
393       screen.state('zoomed')
394       screen.mainloop() #used to close the screen for readfromjson
```

readFromJson() function

a screen is created for this function and we added the attributes or properties needed and we imported the image of the back button and then we have created a button by using this image that calls the main menu at line 352. Then we have opened the file of Json that contains all the information of the students in order to display them all at once and stored them in a variable called “f” then we loaded this file and took the data and stored in another variable called “data”. Then we used TreeView a way to display the data and we have created the columns and give them height and et to be dynamic so that it fits the height based on the given dsata and then we set the order and modified their width. Then we have created position of the whole table in screen. We have created a for loop that passes the name of the object created in json file and prints all the information inside this object in json file every time. Then we will close the file and the screen as well.

```
C: > Users > Elza Morgan > Desktop > project 12th > spl project > Final Section Project > mian.py > ...
383     #function that is used to delete the student
384 def deleteStudent():
385     screen = Tk()
386     screen.title('Registration System')
387     screen.geometry('700x500')
388     screen.config(bg="#447c84")
389 
390     #image of back button
391     img = Image.open("C:/Users/Elza Morgan/Desktop/project 12th/spl project/Untitled-4.png")
392     resized = img.resize((50, 50), Image.ANTIALIAS)
393     newimg = ImageTk.PhotoImage(resized)
394 
395     back = Button(
396         screen,
397         image=newimg,
398         padx=20, pady=10,
399         relief=RAISED,
400         borderwidth=0,
401         background="#447c84",
402         command=lambda:[screen.destroy(),EnterSystem()]
403         )
404     back.place(x=30,y=40)
405 
406     frame = Frame(screen, height=400,width=10,padx=20, pady=20)
407     frame.config(bg="white")
408     frame.pack_propagate(False)
409     frame.pack(expand=True)
410 
411     lbl = Label(
412     frame,
413     text='Delete ID From Database',
```

```
C: > Users > Elza Morgan > Desktop > project 12th > spl project > Final Section Project > mian.py > ...
403         )
404     back.place(x=30,y=40)
405 
406     frame = Frame(screen, height=400,width=10,padx=20, pady=20)
407     frame.config(bg="white")
408     frame.pack_propagate(False)
409     frame.pack(expand=True)
410 
411     lbl = Label(
412     frame,
413     text='Delete ID From Database',
414     font=("Times", "14")
415     ).grid(row=2, column=0, pady=5)
416 
417     value = Entry(frame, width=30) #creating a textfiled
418     value.pack()
419     value.grid(row=2,column=3)
```

deleteStudent() function

We have created a screen for this function and set the suitable properties or attributes for it. We have imported an image which is the back button and then we have created button that lets you go back to the main menu as you call it at line 402 and destroys the current screen. Then we have created a frame inside the screen and inside the frame we have created a label used to delete the student from database by the Id and by it we created a text filed.

```
434     #inside funstion delete student there is another function to apply this action
435     def applyDelete():
436         print(value.get())
437         value1 = int(value.get())#value refered to id as in textfield.
438         try: #used to connect to the database
439             my_connection = mysql.connector.connect(host="localhost",user="root",password="Youyou123@",
440                                                     port ="3306",database="project_database",auth_plugin='mysql_native_password')
441 
442             print("You are connected to the database") #printed in consol for double checking
443             cursor = my_connection.cursor() #used to let us create query in the database
444 
445             #that's the query that it will delete it from the database accoring to the id that will be inserted
446             deleteQuery ="""DELETE FROM students WHERE studentID = %s"""
447 
448             cursor.execute(deleteQuery,(value1,)) # the query
449             my_connection.commit() # this is used to save the changes that have been made to the table
450             messagebox.showinfo(title="Deleted",message="Student has been Delected") #displays a mesage box
451             students = cursor.fetchall()
452 
453             print(cursor.rowcount, "record(s) deleted")
454             for student in students:
455                 print(student)
456 
457             print("User Deleted successfuly")
458 
459             with open('C:/Users/joeel/Desktop/12th project Elza,Youssef, Mohmed and Mohamed/Code/Students_Info.json',"r")as f:
460                 data = json.load(f)
```

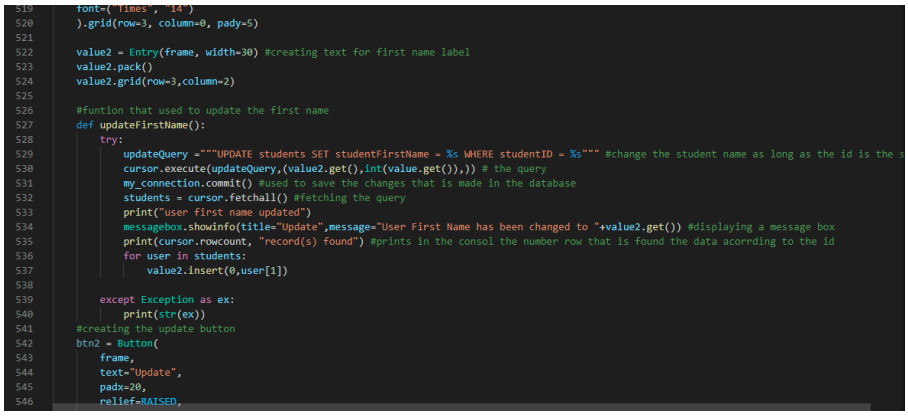
```
459             with open('C:/Users/joeel/Desktop/12th project Elza,Youssef, Mohmed and Mohamed/Code/Students_Info.json',"r")as f:
460                 data = json.load(f)
461                 for student in data["student_details"]:
462                     if student["student_id"] == value.get():
463                         data["student_details"].remove(student)
464             with open('C:/Users/joeel/Desktop/12th project Elza,Youssef, Mohmed and Mohamed/Code/Students_Info.json',"w") as f:
465             # file.seek(0)
466                 json.dump(data, f, indent = 4)
467 
468 
469         except Exception as ex:
470             print(str(ex))
471 
472 
473     #creating button for delete function
474     btn = Button(
475         frame,
476         text="Delete",
477         padx=20, pady=10,
478         relief=RAISED,
479         font=("Times", "16", "bold"),
480         command=applyDelete #calling the deleting function
481         ).grid(row=5, column=1, pady=5)
482 
483     screen.state("zoomed")
484     screen.mainloop()
485     #closing the screen for deleting
486 
```

applyDelete() function

we will access the text field of the Id which is called “value” and we will parse it to int in order to use it and search for in the database. Since text field receive it as a string. We have opened connection with the database and then gave permission in order to write a query. We have created a query that delete the student as long as the id is same as found in the database and then save the table because of the changes that we have made. Then a message box will be displayed as approval of the change that is made. Then we will take each row of the database and store in variable “students” as a list and then print it in the terminal or consol. If it didn’t enter the try then it will display the error message. We have created a button that applies this function when called at line 456.Then we have closed the screen for this function. We can delete the student information from the json file as long as the id is equal to the one found in json file.

```
462 
463     #function used to update any information of that student
464     def updateStudent():
465 
466         try:#opens connection with the database
467             my_connection = mysql.connector.connect(host="localhost",user="root",password="liza2000",
468                                                     port ="3306",database="project_database",auth_plugin='mysql_native_password')
469             print("You are connected to the database")
470             cursor = my_connection.cursor() #enable us to do query in the database
471         except Exception as ex:
472             print(str(ex))
473         #screating screen for the update
474         screen = Tk()
475         screen.title('Registration System')
476         screen.geometry('700x500')
477         screen.config(bg="#447c84")
478 
479         #image of back button
480         img = Image.open("C:/Users/Elza Morgan/Desktop/project 12th/spl project/Untitled-4.png")
481         resized = img.resize((50, 50), Image.ANTIALIAS)
482         newimg = ImageTk.PhotoImage(resized)
483 
484         back = Button(
485             screen,
486             image=newimg,
487             padx=20, pady=10,
488             relief=RAISED,
489             borderwidth=0,
490             background="#447c84",
491             command=lambda:[screen.destroy(),EnterSystem()] #calls the main menu when clicked on
```

```
490         background="#447c84",
491         command=lambda:[screen.destroy(),EnterSystem()] #calls the main menu when clicked on
492         )
493     back.place(x=30,y=40)
494 
495     frame = Frame(screen, height=400,width=10,padx=20, pady=20)
496     frame.config(bg="white")
497     frame.pack_propagate(False)
498     frame.pack(expand=True)
499 
500 
501     #created label for serching by id
502     lbl = Label(
503     frame,
504     text='Search by ID',
505     font=("Times", "14")
506     ).grid(row=2, column=0, pady=5)
507 
508     value = Entry(frame, width=30) #creating a textfield for that label
509     value.pack()
510     value.grid(row=2,column=2)
511 
512 
513 
514     #The upcoming labels are the filds that you can update for
515     #created first name label
516     lbl2 = Label(
517     frame,
518     text='First Name',
519     font=("Times", "14")
520     ).grid(row=3, column=0, pady=5)
```



```
519     font=("Times", "14")
520     ).grid(row=3, column=0, pady=5)
521 
522     value2 = Entry(frame, width=30) #creating text for first name label
523     value2.pack()
524     value2.grid(row=3,column=2)
525 
526     #funtion that used to update the first name
527     def updateFirstName():
528         try:
529             updateQuery ="""UPDATE students SET studentFirstName = %s WHERE studentID = %s""" #change the student name as long as the id is the s
530             cursor.execute(updateQuery,(value2.get(),int(value.get()),)) # the query
531             my_connection.commit() #used to save the changes that is made in the database
532             students = cursor.fetchall() #fetching the query
533             print("user first name updated")
534             messagebox.showinfo(title="Update",message="User First Name has been changed to "+value2.get()) #displaying a message box
535             print(cursor.rowcount, "record(s) found") #prints in the consol the number row that is found the data acorrding to the id
536             for user in students:
537                 value2.insert(0,user[1])
538 
539         except Exception as ex:
540             print(str(ex))
541     #creating the update button
542     btn2 = Button(
543         frame,
544         text="Update",
545         padx=20,
546         relief=RAISED,
```

updateStudent() function

we have opened a connection with the database and created an access to the database in order to write a query in the database. If it didn't create the connection, it will print the error name. Then we have created a screen for this function and we set suitable properties or attributes for it. We have imported an image which is the back button and then we have used it as a button by clicking on it, it will take you to the main menu by calling this function at line 491.Then we have created a frame inside the screen and inside the screen we have created a label for searching this student by the id and then a text field is created by variable called "value". Another label is created for the First name of the student and a text field which is initialized to variable "value2".

```
550     #funtion that used to update the first name
551     def updateFirstName():
552         try:
553             updateQuery ="""UPDATE students SET studentFirstName = %s WHERE studentID = %s""" #change the student name as long as the id is the s
554             cursor.execute(updateQuery,(value2.get(),int(value.get()),)) # the query
555             my_connection.commit() #used to save the changes that is made in the database
556             students = cursor.fetchall() #fetching the query
557             print("user first name updated")
558             messagebox.showinfo(title="Update",message="User First Name has been changed to "+value2.get()) #displaying a message box
559             print(cursor.rowcount, "record(s) found") #prints in the consol the number row that is found the data acorrding to the id
560             #this for loop is used to display the updated text field of first name along with the other data
561             for user in students:
562                 value2.insert(0,user[1])
563             with open('C:/Users/joeel/Desktop/12th project Elza,Youssef, Mohmed and Mohamed/Code/Students_Info.json',"r")as f:
564                 data = json.load(f)
565                 for student in data["student_details"]:
566                     if student["student_id"] == value.get():
567                         student["student_fname"]=value2.get()
568             with open('C:/Users/joeel/Desktop/12th project Elza,Youssef, Mohmed and Mohamed/Code/Students_Info.json',"w") as f:
569             # file.seek(0)
570                 json.dump(data, f, indent = 4)
571
572         except Exception as ex:
573             print(str(ex))
574     #creating the update button
575     btn2 = Button(
576         frame,
```

updateFirstName() function

This function is inside the updatestudent function so we don't have to open the connection. We started directly writing the query, which is as long as the id is same as found in the database update the firstname in the table of students and firstname column. Then display a message box as an approval. Then we have created a for loop that loops over "students" that already stored the updated information about this student and therefore it displays the updated text field along with other information about the students. We have created a button that calls for the function updateFirstname at line 549. Then we have created a label for the lastname and a text field. Moreover, we can update the student first name in json file as well.

```
595     #creating the function that is updated the last name
596     def updateLastName():
597         try:
598             updateQuery ="""UPDATE students SET studentLastName = %s WHERE studentID = %s"""#change the student name as long as the id is the sa
599             cursor.execute(updateQuery,(value3.get(),int(value.get()),)) # the query
600             my_connection.commit() #saves the updated table
601             students = cursor.fetchall() #fetching the query
602             print("user last name updated")
603             messagebox.showinfo(title="Update",message="User Last Name has been changed to "+value3.get()) #display meesage box
604             print(cursor.rowcount, "record(s) found")
605             for user in students:
606                 value2.insert(0,user[2])
607             with open('C:/Users/joeel/Desktop/12th project Elza,Youssef, Mohmed and Mohamed/Code/Students_Info.json',"r")as f:
608                 data = json.load(f)
609                 for student in data["student_details"]:
610                     if student["student_id"] == value.get():
611                         student["student_lname"]=value3.get()
612             with open('C:/Users/joeel/Desktop/12th project Elza,Youssef, Mohmed and Mohamed/Code/Students_Info.json',"w") as f:
613             # file.seek(0)
614                 json.dump(data, f, indent = 4)
615         except Exception as ex:
616             print(str(ex))
617
618     #creats button for lastname
619     btn3 = Button(
620         frame,
621         text="Update"
```

updateLastName() function

This function is inside the updatestudent function so we don't have to open the connection. We started directly writing the query, which is as long as the id is same as found in the database update the lastname in the table of students and lastname column and then save the changes in the table. Then display a message box as an approval. Then we have created a for loop that loops over "students" that already stored the updated information about this student and therefore it displays the updated text field along with other information about the students. We have created a button that calls for the function updateLastName at line 585. Then we have created a label for the email. and a text field. Moreover, we can update the student last name in json file as well.

```
640     #function that is used to update the email
641     def updateEmail():
642         try:
643             updateQuery ="""UPDATE students SET EmailAddress = %s WHERE studentID = %s"""#change the student name as long as the id is the same
644             cursor.execute(updateQuery,(value4.get(),int(value.get()),)) # the query
645             my_connection.commit()#saves the changes in the table
646             students = cursor.fetchall() #fetching the query
647             print("user email updated")
648             messagebox.showinfo(title="Update",message="User Email has been changed to "+value4.get())#display a message box after the action
649             print(cursor.rowcount, "record(s) found")
650 
651             for user in students:
652                 value2.insert(0,user[3])
653             with open('C:/Users/joeel/Desktop/12th project Elza,Youssef, Mohmed and Mohamed/Code/Students_Info.json',"r")as f:
654                 data = json.load(f)
655                 for student in data["student_details"]:
656                     if student["student_id"] == value.get():
657                         student["student_email"]=value4.get()
658             with open('C:/Users/joeel/Desktop/12th project Elza,Youssef, Mohmed and Mohamed/Code/Students_Info.json',"w") as f:
659             # file.seek(0)
660                 json.dump(data, f, indent = 4)
661         except Exception as ex:
662             print(str(ex))
663     #creating button for updating the  email
664     btn4 = Button(
665         frame,
```

updateEmail() function

This function is inside the updatestudent function so we don't have to open the connection. We started directly writing the query, which is as long as the id is same as found in the database update the email in the table of students and email column and then save the changes in the table. Then display a message box as an approval. Then we have created a for loop that loops over "students" that already stored the updated information about this student and therefore it displays the updated text field along with other information about the students. We have created a button that calls for the function updateEmail at line 621. Then we have created a label for the email. and a text field. Moreover, we can update the student email in json file as well.

```
624     #this function is used to apply the search and override the changes that had been made
625     #this functtion perfoms the select query
626     def applySearch():
627         value2.delete(0,'end')
628         value3.delete(0,'end')
629         value4.delete(0,'end')
630 
631         print("ID = "+value.get())
632         value1 = int(value.get()) #takes what's found in textfield of id adn then parse it in order to use it as intger
633         try:
634             SearchQuery ="""SELECT * FROM students WHERE studentID = %s"""
635             cursor.execute(SearchQuery,(value1,)) # enables query in the database
636 
637             students = cursor.fetchall()
638             #used to fill the text field of fname,lname and email with the new data
639             print(cursor.rowcount, "record(s) found")
640             for user in students:
641                 value2.insert(0,user[1])
642                 value3.insert(0,user[2])
643                 value4.insert(0,user[3])
644 
645         except Exception as ex:
646             print(str(ex))
647     #button created for apply search
648     btn = Button(
649         frame,
650         text="Search",
651         padx=20, pady=10,
652         relief=RAISED,
653         font=("Times", "16", "bold"),
654         command=applySearch() #calls the for applyScreach function
```

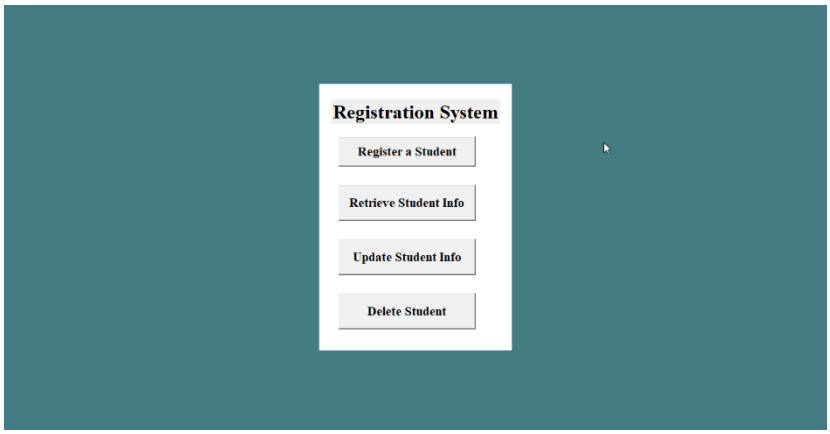
```
644
645         except Exception as ex:
646             print(str(ex))
647     #button created for apply search
648     btn = Button(
649         frame,
650         text="Search",
651         padx=20, pady=10,
652         relief=RAISED,
653         font=("Times", "16", "bold"),
654         command=applySearch() #calls the for applyScreach function
655         ).grid(row=6, column=3, pady=5)
656
657     screen.state("zoomed")
658     screen.mainloop()
659 #closes the screen for the update studnet function
660
661 #starts the whole file that we are at
662 if __name__ =="__main__":
663     MainScreen()
664
665
```

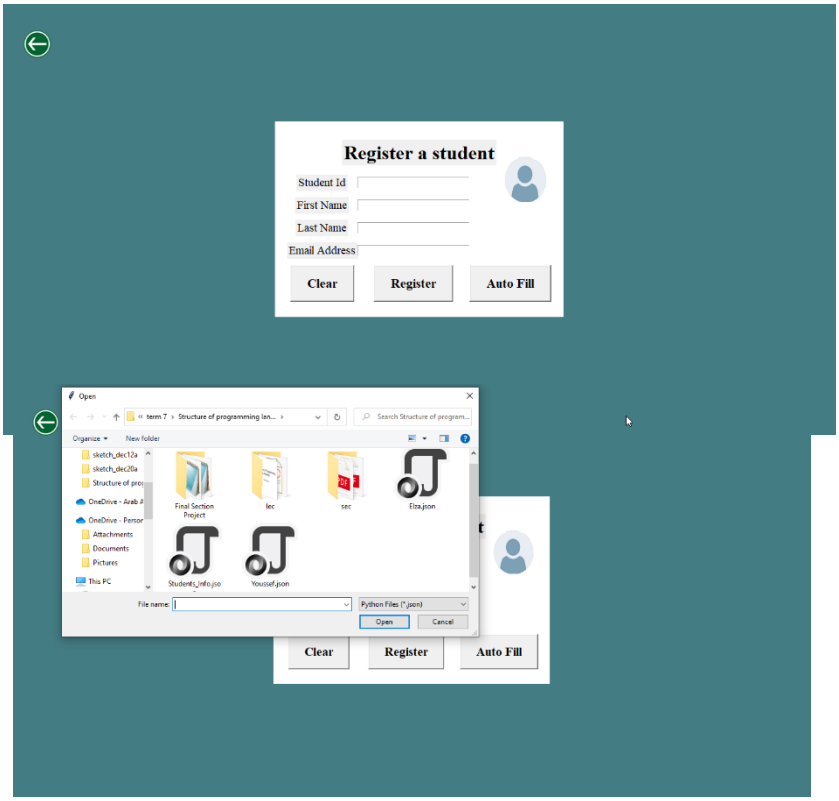
applySearch() function

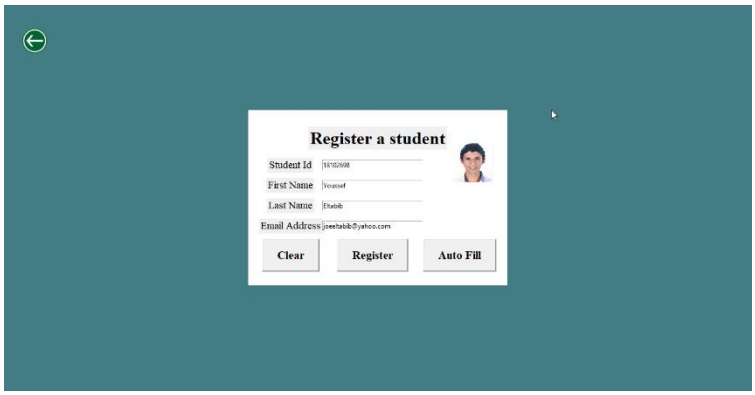
used to delete everything in the text field and then displays everything on the text field as long as the id is found in the database. We have created the button that will be called by line 654 in order go to applySearch function. Then we have closed the screen for updating.

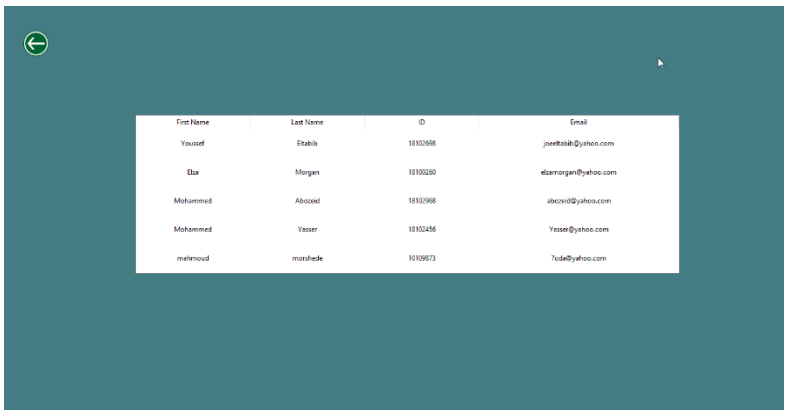
Line 662 and 663 is used to make the whole file or code start and display the main screen as the code runs and the GUI appears.

Images of our Application Running

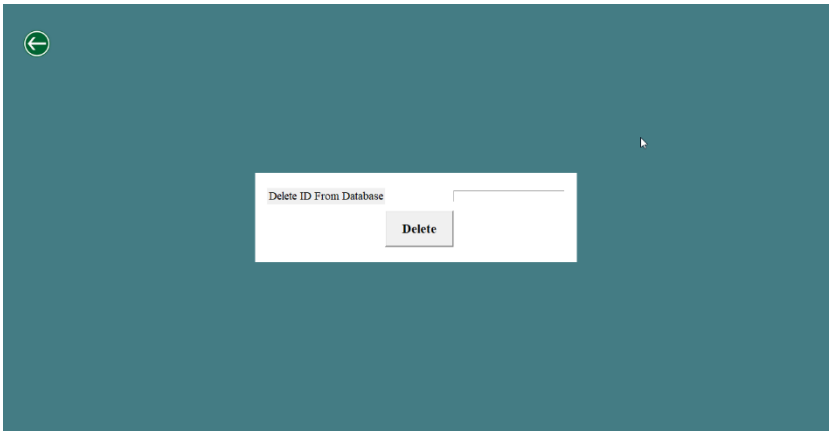



Register a student
Student Id
First Name
Last Name
Email Address
Clear
Register
Auto Fill
Open
Organize
New folder
Final Section Project
lec
sec
Eiza.json
Students_Info.jso
Youssef.json
File name:
Python Files (*.json)
Open
Cancel


Register a student
Student Id
First Name
Youssef
Last Name
Eltabib
Email Address
joeeltabib@yahoo.com
Clear
Register
Auto Fill


First Name
Last Name
ID
Email
Youssef
Eltabib
joeeltabib@yahoo.com
Eiza
Morgan
elzamorgan@yahoo.com
Mohammed
Abozeid
abozeid@yahoo.com
Mohammed
Yasser
Yasser@yahoo.com
mahmoud
morshede
7ede@yahoo.com


Search by ID
First Name
Update
Last Name
Update
Email
Update
Search


Delete ID From Database
Delete