```python
from tkinter import *
from tkinter import ttk
from PIL import Image,ImageTk
import os
import pickle
import mysql.connector  as sql
from tkinter import messagebox
from datetime import date
from datetime import time
from datetime import *
import requests
from bs4 import BeautifulSoup
import time
import user_inter
import csv


def regist_stu():
    def validation():
        """this will validate if the username and email of entry fields are already in database table named student or
        not if return True, error message is thrown displaying email/username already exists"""
        try:
            #obj_student_database = Model_class.student_registration.GetDatabase('use cms;')
            #db_connection.create(obj_student_database.get_database())
            a=load_data()
            host=a[0]
            username = a[2]
            password = a[3]
            port=a[1]

            spec=sql.connect(host=host,user=username,password=password,port=port,database="sms")
            mycur=spec.cursor()

            query = "select * from students;"
            mycur.execute(query)
            data = mycur.fetchall()
            # print(data)
            username_list = []
            email_list = []
            for values in data:
                # print(values)
                user_data_list = values[1]
                username_list.append(user_data_list)
                email_data_list = values[2]
                email_list.append(email_data_list)
                # print(final_list)
                # print(data_list)
        except BaseException as msg:
            print(msg)

        if username_entry.get() == "" or email_entry.get() == "" or password_entry.get() == "" \
                or f_name_entry.get() == "" or l_name_entry.get() == "" or dob_entry.get() == "" \
                or address_entry.get() == "" or contact_entry.get() == "":
            messagebox.showwarning("Warning", "All Fields are Required\n Please fill all required fields")

        elif username_entry.get() in username_list:
            messagebox.showerror("Already Exists", f"{username_entry.get()} username Already Exists")
```

```
58
59          elif email_entry.get() in email_list:
60              messagebox.showerror("Already Exists", f"{email_entry.get()} Email ID Already
    Exists")
61
62          elif password_entry.get() != c_password_entry.get():
63              messagebox.showerror("Not Matched", "Password Does not Matched")
64
65          else:
66              click_submit()
67
68      def click_submit():
69          """initialize when click submit button, which will take data from entry box
70          and insert those data into student table after successful validation of those data"""
71          try:
72              #obj_student_database = Model_class.student_registration.GetDatabase('use cms;')
73              #db_connection.create(obj_student_database.get_database())
74              a=load_data()
75              host=a[0]
76              username = a[2]
77              password = a[3]
78              port=a[1]
79
80              spec=sql.connect(host=host,user=username,password=password,port=port,database="sms
    ")
81              mycur=spec.cursor()
82
83              #obj_student_database = Model_class.student_registration.StudentRegistration(
    username_entry.get(),
84                                                                                          #
    email_entry.get(),
85                                                                                          #
    password_entry.get(),
86                                                                                          #
    f_name_entry.get(),
87                                                                                          #
    l_name_entry.get(),
88                                                                                          #
    dob_entry.get(),
89                                                                                          #
    gender_combo.get(),
90                                                                                          #
    address_entry.get(),
91                                                                                          #
    contact_entry.get(),
92                                                                                          #
    shift_combo.get(),
93                                                                                          #
    course_combo.get(),
94                                                                                          #
    batch_combo.get(),
95                                                                                          #
    section_combo.get(),
96                                                                                          #
    reg_date)
97              query = f"insert into students (username,email,password,f_name,l_name,dob,gender,
    address," \
98                      f"contact_no,shift,course_enrolled,batch,section_enrolled,reg_date) values
     ('{username_entry.get()}','{email_entry.get()}','{password_entry.get()}','{f_name_entry.get()
    }','{l_name_entry.get()}','{dob_entry.get()}'," \
99                      f"'{gender_combo.get()}','{address_entry.get()}','{contact_entry.get()}','
    {shift_combo.get()}','{course_combo.get()}','{batch_combo.get()}','{section_combo.get()}','{
    reg_date}');"
```

```
100             #values = (obj_student_database.get_username(), obj_student_database.get_email(),
101                        #obj_student_database.get_password(), obj_student_database.get_f_name
    (),
102                        #obj_student_database.get_l_name(), obj_student_database.get_dob(),
103                        #obj_student_database.get_gender(), obj_student_database.get_address
    (),
104                        #obj_student_database.get_contact(), obj_student_database.get_shift(),
105                        #obj_student_database.get_course_id(), obj_student_database.
    get_batch_id(),
106                        #obj_student_database.get_section(), obj_student_database.get_reg_date
    ())
107             # print(values)
108             mycur.execute(query)
109             spec.commit()
110             # print(values)
111             messagebox.showinfo("Success", f"Data inserted Successfully\n Name={username_entry
    .get()},\n "
112                                             f"registration={reg_date}")
113
114         except BaseException as msg:
115             print(msg)
116             messagebox.showerror("Error", "There is some error Submitting Credentials ")
117
118     def click_clear_button():
119         """this will clear entire field to default when click on clear button"""
120         username_entry.delete(0, END)
121         f_name_entry.delete(0, END)
122         l_name_entry.delete(0, END)
123         email_entry.delete(0, END)
124         password_entry.delete(0, END)
125         c_password_entry.delete(0, END)
126         dob_entry.delete(0, END)
127         gender_combo.current(0)
128         address_entry.delete(0, END)
129         contact_entry.delete(0, END)
130         shift_combo.current(0)
131         batch_combo.current(0)
132         course_combo.current(0)
133         batch_combo.current(0)
134         section_combo.current(0)
135
136     def back():
137         root.destroy()
138
139     def exit():
140         ask = messagebox.askyesnocancel("Confirm Exit", "Are you sure you want to Exit\n
    Student Registration Form?")
141         if (ask == True):
142             root.destroy()
143
144     root = Toplevel()
145     root.title('SCHOOL MANAGEMENT SYSTEM')
146     root.geometry('1067x600')
147     root.config(bg="#f29844")
148     #root.iconbitmap('images\\logo.ico')
149     root.resizable(False, False)
150     #manage_student_frame = ImageTk.PhotoImage(file='Pics\\student_frame.png')
151
152     # ======================Variables=======================
153
154     reg_frame = Frame(root, bg="#ffffff", width=1000, height=560)
155     reg_frame.place(x=30, y=30)
156
```

```
157     heading = Label(reg_frame, text="Student Registration Form", font=('yu gothic ui', 20, "
    bold"), bg="white",
158                             fg='black',
159                             bd=5,
160                             relief=FLAT)
161     heading.place(x=200, y=0, width=600)
162 
163 
164     cred_frame = LabelFrame(reg_frame, text="Account Details", bg="white", fg="#4f4e4d",
    height=140,
165                                     width=800, borderwidth=2.4,
166                                     font=("yu gothic ui", 13, "bold"))
167     cred_frame.config(highlightbackground="red")
168     cred_frame.place(x=100, y=50)
169 
170     # ============================================================================
171     # ============================Username========================================
172     # ============================================================================
173 
174     username_label = Label(cred_frame, text="Username ", bg="white", fg="#4f4e4d",
175                                 font=("yu gothic ui", 13, "bold"))
176     username_label.place(x=10, y=10)
177 
178     username_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69",
179                                 font=("yu gothic ui semibold", 12))
180     username_entry.place(x=230, y=117, width=260)  # trebuchet ms
181 
182     username_line = Canvas(root, width=260, height=1.5, bg="#bdb9b1", highlightthickness=0)
183     username_line.place(x=230, y=140)
184 
185     # ============================================================================
186     # ============================Email===========================================
187     # ============================================================================
188 
189     email_label = Label(cred_frame, text="Email ", bg="white", fg="#4f4e4d",
190                                 font=("yu gothic ui", 13, "bold"))
191     email_label.place(x=370, y=10)
192 
193     email_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69",
194                                 font=("yu gothic ui semibold", 12))
195     email_entry.place(x=555, y=117, width=350)  # trebuchet ms
196 
197     email_line = Canvas(root, width=350, height=1.5, bg="#bdb9b1", highlightthickness=0)
198     email_line.place(x=555, y=140)
199 
200     # ============================================================================
201     # ============================Password========================================
202     # ============================================================================
203 
204     password_label = Label(cred_frame, text="Password ", bg="white", fg="#4f4e4d",
205                                 font=("yu gothic ui", 13, "bold"))
206     password_label.place(x=10, y=50)
207 
208     password_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69",
209                                 font=("yu gothic ui semibold", 12), show="*")
210     password_entry.place(x=230, y=157, width=260)  # trebuchet ms
211 
212     password_line = Canvas(root, width=260, height=1.5, bg="#bdb9b1", highlightthickness=0)
213     password_line.place(x=230, y=180)
214 
215     # ============================================================================
216     # ========================Confirm password====================================
217     # ============================================================================
```

```
218
219     c_password_label = Label(cred_frame, text="Confirm Password ", bg="white", fg="#4f4e4d",
220                                      font=("yu gothic ui", 13, "bold"))
221     c_password_label.place(x=370, y=50)
222
223     c_password_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69"
    ,
224                                      font=("yu gothic ui semibold", 12), show="*")
225     c_password_entry.place(x=650, y=157, width=255)  # trebuchet ms
226
227     c_password_line = Canvas(root, width=255, height=1.5, bg="#bdb9b1", highlightthickness=0)
228     c_password_line.place(x=650, y=180)
229
230
231     # ============================================================================
232     # ========================frame for personal credentials ================
233     # ============================================================================
234
235     personal_frame = LabelFrame(reg_frame, text="Personal Details", bg="white", fg="#4f4e4d",
    height=265,
236                                         width=800, borderwidth=2.4,
237                                         font=("yu gothic ui", 13, "bold"))
238     personal_frame.config(highlightbackground="red")
239     personal_frame.place(x=100, y=200)
240
241
242
243     # ============================================================================
244     # ============================First name=====================================
245     # ============================================================================
246     f_name_label = Label(personal_frame, text="First Name ", bg="white", fg="#4f4e4d",
247                                  font=("yu gothic ui", 13, "bold"))
248     f_name_label.place(x=10, y=10)
249
250     f_name_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69",
251                                  font=("yu gothic ui semibold", 12))
252     f_name_entry.place(x=235, y=267, width=260)  # trebuchet ms
253
254     f_name_line = Canvas(root, width=260, height=1.5, bg="#bdb9b1", highlightthickness=0)
255     f_name_line.place(x=235, y=290)
256
257     # ============================================================================
258     # ============================Last name=====================================
259     # ============================================================================
260
261     l_name_label = Label(personal_frame, text="Last Name ", bg="white", fg="#4f4e4d",
262                                  font=("yu gothic ui", 13, "bold"))
263     l_name_label.place(x=370, y=10)
264
265     l_name_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69",
266                                  font=("yu gothic ui semibold", 12))
267     l_name_entry.place(x=595, y=267, width=315)  # trebuchet ms
268
269     l_name_line = Canvas(root, width=315, height=1.5, bg="#bdb9b1", highlightthickness=0)
270     l_name_line.place(x=595, y=290)
271
272     # ============================================================================
273     # ============================DOB=============================================
274     # ============================================================================
275
276     dob_label = Label(personal_frame, text="DOB ", bg="white", fg="#4f4e4d",
277                               font=("yu gothic ui", 13, "bold"))
278     dob_label.place(x=10, y=50)
```

```
279
280     dob_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69",
281                               font=("yu gothic ui semibold", 12))
282     dob_entry.insert(0, "mm/dd/yyyy")
283     dob_entry.place(x=190, y=307, width=305)  # trebuchet ms
284     #dob_entry.bind("<1>", pick_date)
285
286     dob_line = Canvas(root, width=305, height=1.5, bg="#bdb9b1", highlightthickness=0)
287     dob_line.place(x=190, y=329)
288
289     # ============================================================================
290     # ============================Gender==========================================
291     # ============================================================================
292     style = ttk.Style()
293
294     # style.map('TCombobox', selectbackground=[('readonly', 'grey')])
295     root.option_add("*TCombobox*Listbox*Foreground", '#f29844')
296
297     gender_label = Label(personal_frame, text="Gender ", bg="white", fg="#4f4e4d",
298                                  font=("yu gothic ui", 13, "bold"))
299     gender_label.place(x=370, y=50)
300
301     gender_combo = ttk.Combobox(root, font=('yu gothic ui semibold', 12, 'bold'), state='
    readonly',
302                                           width=35)
303
304     gender_list = ['Male', 'Female', 'Rather not say']
305     gender_combo['values'] = gender_list
306     # gender_combo.current(0)
307     gender_combo.place(x=570, y=307)
308
309     # gender_line = Canvas(root, width=315, height=1.5, bg="#bdb9b1", highlightthickness=0)
310     # gender_line.place(x=595, y=369)
311
312     # ============================================================================
313     # ============================Address=========================================
314     # ============================================================================
315
316     address_label = Label(personal_frame, text="Address ", bg="white", fg="#4f4e4d",
317                                  font=("yu gothic ui", 13, "bold"))
318     address_label.place(x=10, y=90)
319
320     address_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69",
321                                  font=("yu gothic ui semibold", 12))
322     address_entry.place(x=215, y=347, width=280)  # trebuchet ms
323
324     address_line = Canvas(root, width=280, height=1.5, bg="#bdb9b1", highlightthickness=0)
325     address_line.place(x=215, y=370)
326
327     # ============================================================================
328     # ============================Contact no======================================
329     # ============================================================================
330
331     contact_label = Label(personal_frame, text="Contact No. ", bg="white", fg="#4f4e4d",
332                                  font=("yu gothic ui", 13, "bold"))
333     contact_label.place(x=370, y=90)
334
335     contact_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69",
336                                  font=("yu gothic ui semibold", 12))
337     contact_entry.place(x=605, y=347, width=305)  # trebuchet ms
338
339     contact_line = Canvas(root, width=305, height=1.5, bg="#bdb9b1", highlightthickness=0)
340     contact_line.place(x=605, y=370)
```

```
341
342     # ========================================================================
343     # ============================Shift No====================================
344     # ========================================================================
345
346     shift_label = Label(personal_frame, text="Shift ", bg="white", fg="#4f4e4d",
347                                 font=("yu gothic ui", 13, "bold"))
348     shift_label.place(x=10, y=130)
349
350     shift_combo = ttk.Combobox(root, font=('yu gothic ui semibold', 12, 'bold'), state='
    readonly',
351                                     width=28)
352     shift_list = ["Morning", "Day", "Evening"]
353     shift_combo['values'] = shift_list
354     shift_combo.current(0)
355     shift_combo.place(x=213, y=387)
356
357     def load_data():
358         f=open("Credentials.csv","r")
359         s=csv.reader(f,delimiter="-")
360         d=[]
361         for i in s:
362             d.append(i)
363         a=d[::-1]
364         return (a[0])
365
366     try:
367         #obj_student_database = Model_class.student_registration.GetDatabase('use cms;')
368         #db_connection.create(obj_student_database.get_database())
369         a=load_data()
370         host=a[0]
371         username = a[2]
372         password = a[3]
373         port=a[1]
374
375
376         spec=sql.connect(host=host,user=username,password=password,port=port,database="sms")
377         mycur=spec.cursor()
378
379         query = "select * from section"
380         mycur.execute(query)
381         #data=mycur.fetchall()
382         query = "select * from section"
383         section_tuple = mycur.fetchall()
384         # print(section_tuple)
385         section_list = []
386         for i in section_tuple:
387             section_name = i[2]
388             section_list.append(section_name)
389             # print(section_name)
390             # print(section_list)
391
392     except BaseException as msg:
393         print(msg)
394
395     try:
396         #obj_student_database = Model_class.student_registration.GetDatabase('use cms;')
397         #db_connection.create(obj_student_database.get_database())
398         a=load_data()
399         host=a[0]
400         username = a[2]
401         password = a[3]
402         port=a[1]
```

```
403
404
405         spec=sql.connect(host=host,user=username,password=password,port=port,database="sms")
406         mycur=spec.cursor()
407
408         query = "select * from course"
409         mycur.execute(query)
410         course_tuple = mycur.fetchall()
411
412         # print(course_tuple)
413         course_list = []
414         for i in course_tuple:
415             course_name = i[1]
416             course_list.append(course_name)
417             # print(course_name)
418             # print(course_list)
419
420     except BaseException as msg:
421         print(msg)
422
423     try:
424         #obj_student_database = Model_class.student_registration.GetDatabase('use cms;')
425         #db_connection.create(obj_student_database.get_database())
426         a=load_data()
427         host=a[0]
428         username = a[2]
429         password = a[3]
430         port=a[1]
431
432
433         spec=sql.connect(host=host,user=username,password=password,port=port,database="sms")
434         mycur=spec.cursor()
435
436         query = "select * from batch"
437         mycur.execute(query)
438         batch_tuple = mycur.fetchall()
439         # print(batch_tuple)
440         batch_list = []
441         for i in batch_tuple:
442             batch_name = i[1]
443             batch_list.append(batch_name)
444             # print(batch_name)
445             # print(batch_list)
446
447     except BaseException as msg:
448         print(msg)
449
450
451
452
453
454
455
456
457
458     # ============================================================================
459     # ============================Course enrolled=================================
460     # ============================================================================
461
462     course_label = Label(personal_frame, text="Course Enrolled ", bg="white", fg="#4f4e4d",
463                                  font=("yu gothic ui", 13, "bold"))
464     course_label.place(x=370, y=130)
465
```

```
466     course_combo = ttk.Combobox(root, font=('yu gothic ui semibold', 12, 'bold'), state='
    readonly',
467                                         width=28)
468 
469     course_combo['values'] = course_list
470     try:
471         course_combo.current(0)
472     except:
473         messagebox.showerror("Error","You must add course first")
474     course_combo.place(x=635, y=387)
475 
476     # ========================================================================
477     # ============================Batch=======================================
478     # ========================================================================
479 
480     batch_label = Label(personal_frame, text="Batch ", bg="white", fg="#4f4e4d",
481                                 font=("yu gothic ui", 13, "bold"))
482     batch_label.place(x=10, y=170)
483 
484     batch_combo = ttk.Combobox(root, font=('yu gothic ui semibold', 12, 'bold'), state='
    readonly',
485                                     width=28)
486 
487     batch_combo['values'] = batch_list
488     try:
489         batch_combo.current(0)
490     except:
491         messagebox.showerror("Error", "You must add batch first")
492     batch_combo.place(x=213, y=427)
493 
494     # ========================================================================
495     # ============================Section=====================================
496     # ========================================================================
497 
498     section_label = Label(personal_frame, text="Section ", bg="white", fg="#4f4e4d",
499                                 font=("yu gothic ui", 13, "bold"))
500     section_label.place(x=370, y=170)
501 
502     section_combo = ttk.Combobox(root, font=('yu gothic ui semibold', 12, 'bold'), state='
    readonly',
503                                         width=28)
504 
505     section_combo['values'] = section_list
506     try:
507         section_combo.current(0)
508     except:
509         messagebox.showerror("Error", "You must add section first")
510     section_combo.place(x=635, y=427)
511 
512     reg_date = time.strftime("%Y/%m/%d")
513 
514     # ========================================================================
515     # ============================Register options============================
516     # ========================================================================
517 
518     options_frame = LabelFrame(reg_frame, text="Register Options", bg="white", fg="#4f4e4d",
    height=83,
519                                     width=800, borderwidth=2.4,
520                                     font=("yu gothic ui", 13, "bold"))
521     options_frame.config(highlightbackground="red")
522     options_frame.place(x=100, y=470)
523 
524     # ========================================================================
```

```
525     # ============================Register options=====================================
526     # ================================================================================
527     submit_img = ImageTk.PhotoImage \
528         (file='Pics\\submit.png')
529 
530     submit = Button(options_frame, image=submit_img,
531                             font=("yu gothic ui", 13, "bold"), relief=FLAT, activebackground="
    white"
532                             , borderwidth=0, background="white", cursor="hand2"
533                             , command=validation)
534 
535     submit.image = submit_img
536     submit.place(x=90, y=10)
537 
538     clear_img = ImageTk.PhotoImage \
539         (file='Pics\\clear.png')
540     clear_button = Button(options_frame, image=clear_img,
541                                 font=("yu gothic ui", 13, "bold"), relief=FLAT,
    activebackground="white"
542                                 , borderwidth=0, background="white", cursor="hand2"
543                                 ,command=click_clear_button)
544     clear_button.image = clear_img
545     clear_button.place(x=250, y=13)
546 
547     back_img = ImageTk.PhotoImage \
548         (file='Pics\\back.png')
549     back_button = Button(options_frame, image=back_img,
550                                 font=("yu gothic ui", 13, "bold"), relief=FLAT,
    activebackground="white"
551                                 , borderwidth=0, background="white", cursor="hand2"
552                                 , command=back)
553     back_button.image = back_img
554     back_button.place(x=410, y=13)
555 
556     exit_img = ImageTk.PhotoImage \
557         (file='Pics\\exit.png')
558     exit_button = Button(options_frame, image=exit_img,
559                                 font=("yu gothic ui", 13, "bold"), relief=FLAT,
    activebackground="white"
560                                 , borderwidth=0, background="white", cursor="hand2", command=
    exit)
561     exit_button.image = exit_img
562     exit_button.place(x=570, y=13)
563 
564 
565     #root.mainloop()
566 
567 if __name__ =="__main__":
568     regist_stu()
```