```python
from distutils import command
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
import course_screen


def regist_batch():
    def click_clear_button():
        batch_name_entry.delete(0, END)
        batch_year_entry.delete(0, END)
        batch_intake_combo.current(0)

    def back():
        root.destroy()

    def load_data():
        f=open("Credentials.csv","r")
        s=csv.reader(f,delimiter="-")
        d=[]
        for i in s:
            d.append(i)
        a=d[::-1]
        return (a[0])

    def validation():
        """this will validate if the batch code and name of entry fields are already in database table named
        batch or not if return True, error message is thrown displaying batch code/name already exists"""
        try:
            #obj_batch_database = Model_class.batch_registration.GetDatabase('use cms;')
            #db_connection.create(obj_batch_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 batch;"
            mycur.execute(query)
            data = mycur.fetchall()

            # print(data)
            name_list = []
            for values in data:
                name_data_list = values[1]
```

```
61                 name_list.append(name_data_list)
62                 # print(name_data_list)
63 
64         except BaseException as msg:
65             print(msg)
66 
67         if batch_name_entry.get() == "" or batch_year_entry.get() == "":
68             messagebox.showwarning("Warning", "All Fields are Required\n Please fill all
   required fields")
69 
70         elif batch_name_entry.get() in name_list:
71             messagebox.showerror("Already Exists", f"{batch_name_entry.get()} Batch Already
   Exists")
72 
73         else:
74             click_submit()
75 
76     def click_submit():
77         """initialize when click submit button, which will take data from entry box
78         and insert those data into student table after successful validation of those data"""
79         try:
80             #obj_batch_database = Model_class.batch_registration.GetDatabase('use cms;')
81             #db_connection.create(obj_batch_database.get_database())
82             a=load_data()
83             host=a[0]
84             username = a[2]
85             password = a[3]
86             port=a[1]
87 
88             spec=sql.connect(host=host,user=username,password=password,port=port,database="sms
   ")
89             mycur=spec.cursor()
90 
91             #obj_batch_database = Model_class.batch_registration.BatchRegistration(
   batch_name_entry.get(),
92                                                                                 #
   batch_year_entry.get(),
93                                                                                 #
   batch_intake_combo.get(),
94                                                                                 #reg_date)
95 
96             query = f"insert into batch (batch_name,batch_year,batch_intake,reg_date) values
    ('{batch_name_entry.get()}','{batch_year_entry.get()}','{batch_intake_combo.get()}','{
   reg_date}');"
97             mycur.execute(query)
98             spec.commit()
99             #values = (obj_batch_database.get_name(),obj_batch_database.get_year(),
100                         #obj_batch_database.get_intake(),obj_batch_database.get_reg_date())
101            # print(values)
102            #db_connection.insert(query, values)
103            # print(values)
104 
105            messagebox.showinfo("Success", f"Batch Registered Successfully\n Batch Name={
   batch_name_entry.get()},\n "
106                                            f"Batch Year={batch_year_entry.get()}")
107 
108        except BaseException as msg:
109            print(msg)
110            messagebox.showerror("Error", "There is some error Submitting Credentials ")
111 
112    def exit():
113        ask = messagebox.askyesnocancel("Confirm Exit", "Are you sure you want to Exit\n
   College Management System?")
```

```
114         if ask is True:
115             root.destroy()
116 
117     root = Toplevel()
118     root.title('BATCH REGISTRATION FORM - COLLEGE MANAGEMENT SYSTEM')
119     root.geometry('1067x600')
120     root.config(bg="#f29844")
121     root.resizable(False, False)
122 
123 
124     # ======================Backend connection=============
125     #db_connection = Backend.connection.DatabaseConnection()
126 
127     # creating frame for Register
128     # img = img
129     # dummylabel = Label(root, image=img)
130     # dummylabel.place(x=30, y=30)
131 
132     reg_frame = Frame(root, bg="#ffffff", width=1000, height=560)
133     reg_frame.place(x=30, y=30)
134 
135 
136 
137     heading = Label(reg_frame, text="Batch Registration Form", font=('yu gothic ui', 20, "bold
    "), bg="white",
138                             fg='black',
139                             bd=5,
140                             relief=FLAT)
141     heading.place(x=200, y=0, width=600)
142 
143 
144 
145     batch_frame = LabelFrame(reg_frame, text="Batch Details", bg="white", fg="#4f4e4d", height
    =380,
146                                     width=800, borderwidth=2.4,
147                                     font=("yu gothic ui", 13, "bold"))
148     batch_frame.config(highlightbackground="red")
149     batch_frame.place(x=100, y=90)
150 
151     # ============================================================================
152     # ============================batch Name=====================================
153     # ============================================================================
154 
155     batch_name_label = Label(batch_frame, text="Batch Name ", bg="white", fg="#4f4e4d",
156                                     font=("yu gothic ui", 13, "bold"))
157     batch_name_label.place(x=160, y=65)
158 
159     batch_name_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69"
    ,
160                                     font=("yu gothic ui semibold", 12))
161     batch_name_entry.place(x=400, y=212, width=345)  # trebuchet ms
162 
163     batch_name_line = Canvas(root, width=345, height=1.5, bg="#bdb9b1", highlightthickness=0)
164     batch_name_line.place(x=400, y=234)
165 
166     # ============================================================================
167     # ============================batch YEAR ====================================
168     # ============================================================================
169     date = time.strftime("%Y")
170 
171     batch_year_label = Label(batch_frame, text="Batch Year ", bg="white", fg="#4f4e4d",
172                                     font=("yu gothic ui", 13, "bold"))
173     batch_year_label.place(x=160, y=115)
```

```
174
175     batch_year_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69"
    ,
176                                      font=("yu gothic ui semibold", 12))
177     batch_year_entry.place(x=390, y=262, width=355)  # trebuchet ms
178
179     batch_year_line = Canvas(root, width=355, height=1.5, bg="#bdb9b1", highlightthickness=0)
180     batch_year_line.place(x=390, y=284)
181     batch_year_entry.insert(0, date)
182
183     # ============================================================================
184     # ============================batch Intake=================================
185     # ============================================================================
186     root.option_add("*TCombobox*Listbox*Foreground", '#f29844')
187
188     batch_intake_label = Label(batch_frame, text="Batch Intake ", bg="white", fg="#4f4e4d",
189                                      font=("yu gothic ui", 13, "bold"))
190     batch_intake_label.place(x=160, y=165)
191
192     batch_intake_combo = ttk.Combobox(batch_frame, font=('yu gothic ui semibold', 12, 'bold'),
193                                               state='readonly',
194                                               width=35)
195     batch_intake_combo['values'] = ("January", "February", "March", "April", "May", "June", "
    July", "August",
196                                               "September", "October", "November", "December")
197     batch_intake_combo.current(0)
198     batch_intake_combo.place(x=270, y=167)
199     # batch_intake_line.place(x=410, y=424)
200
201     reg_date = time.strftime("%Y/%m/%d")
202
203
204     # ============================================================================
205     # ============================Register options==================================
206     # ============================================================================
207     submit_img = ImageTk.PhotoImage(file='Pics\\submit.png')
208     submit = Button(batch_frame, image=submit_img,
209                               font=("yu gothic ui", 13, "bold"), relief=FLAT, activebackground="
    white"
210                               , borderwidth=0, background="white", cursor="hand2",command=
    validation)
211     submit.image = submit_img
212     submit.place(x=90, y=267)
213
214     clear_img = ImageTk.PhotoImage(file='Pics\\clear.png')
215     clear_button = Button(batch_frame, image=clear_img,
216                                   font=("yu gothic ui", 13, "bold"), relief=FLAT,
    activebackground="white"
217                                   , borderwidth=0, background="white", cursor="hand2",
218                                   command=click_clear_button)
219     clear_button.image = clear_img
220     clear_button.place(x=250, y=270)
221
222     back_img = ImageTk.PhotoImage(file='Pics\\back.png')
223     back_button = Button(batch_frame, image=back_img,
224                                   font=("yu gothic ui", 13, "bold"), relief=FLAT,
    activebackground="white"
225                                   , borderwidth=0, background="white", cursor="hand2",command=
    back)
226     back_button.image = back_img
227     back_button.place(x=410, y=270)
228
229     exit_img = ImageTk.PhotoImage(file='Pics\\exit.png')
```

```
230     exit_button = Button(batch_frame, image=exit_img,
231                                  font=("yu gothic ui", 13, "bold"), relief=FLAT,
    activebackground="white"
232                                  , borderwidth=0, background="white", cursor="hand2", command=
    exit)
233     exit_button.image = exit_img
234     exit_button.place(x=570, y=270)
235 
236 
237     #root.mainloop()
238 
239 
240 if __name__ =="__main__":
241     regist_batch()
```