```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 course_reg():
    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 click_submit():
        """initialize when click submit button, which will take data from entry box
        and insert those data into student table after successful validation of those data"""
        try:
            #obj_course_database = Model_class.course_registration.GetDatabase('use cms;')
            #self.db_connection.create(obj_course_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()
            #obj_course_database = Model_class.course_registration.CourseRegistration(self.course_name_entry.get(),
                                                                                      #self.course_duration_entry.get(),
                                                                                      #self.course_credit_entry.get(),
                                                                                      #self.reg_date)

            cn=course_name_entry.get()
            cd=course_duration_entry.get()
            cc=course_credit_entry.get()

            query = f"insert into course (course_name,course_duration,course_credit,reg_date) values ('{cn}','{cd}','{cc}','{reg_date}');"
            mycur.execute(query)
            spec.commit()

            mycur.execute("select * from course;")
```

```
58              value = mycur.fetchall()
59              print(value)
60              # print(values)
61              #self.db_connection.insert(query, values)
62              # print(values)
63              messagebox.showinfo("Success", f"Data inserted Successfully\n Course name={
    course_name_entry.get()}")
64              course_screen.click_view_all()
65
66          except BaseException as msg:
67              print(msg)
68              messagebox.showerror("Error", f"There is some error Submitting Credentials")
69
70      def back():
71          root.destroy()
72
73      def validation():
74          """this will validate if the course code and name of entry fields are already in
    database table named
75          course or not if return True, error message is thrown displaying course code/name
    already exists"""
76          name_list = []
77
78          try:
79              #obj_course_database = Model_class.course_registration.GetDatabase('use cms;')
80              #self.db_connection.create(obj_course_database.get_database())
81              a=load_data()
82              host=a[0]
83              username = a[2]
84              password = a[3]
85              port=a[1]
86
87              spec=sql.connect(host=host,user=username,password=password,port=port,database="sms
    ")
88              mycur=spec.cursor()
89              query = "select * from course;"
90              mycur.execute(query)
91              spec.commit()
92              data = mycur.fetchall()
93
94              # print(data)
95
96              name_list = []
97              for values in data:
98                  name_data_list = values[1]
99                  name_list.append(name_data_list)
100
101         except BaseException as msg:
102             print(msg)
103
104         if course_name_entry.get() == "" or course_duration_entry.get() == "" or \
105                 course_credit_entry.get() == "" :
106             messagebox.showwarning("Warning", "All Fields are Required\n Please fill all
    required fields")
107
108         elif course_name_entry.get() in name_list:
109             messagebox.showerror("Already Exists", f"{course_name_entry.get()} Course Already
    Exists")
110
111         else:
112             click_submit()
113
114
```

```
115
116     root = Toplevel()
117     root.title('COURSE REGISTRATION FORM - COLLEGE MANAGEMENT SYSTEM')
118     root.geometry('1067x600')
119     root.config(bg="#f29844")
120     #root.iconbitmap('images\\logo.ico')
121     root.resizable(False, False)
122     #manage_student_frame = ImageTk.PhotoImage(file='Pics\\student_frame.png')
123
124
125     # =====================Variables======================
126
127     reg_frame = Frame(root, bg="#ffffff", width=1000, height=560)
128     reg_frame.place(x=30, y=30)
129
130     heading = Label(reg_frame, text="Course Registration Form", font=('yu gothic ui', 20, "
    bold"), bg="white",
131                             fg='black',
132                             bd=5,
133                             relief=FLAT)
134     heading.place(x=200, y=0, width=600)
135
136
137     course_frame = LabelFrame(reg_frame, text="Course Details", bg="white", fg="#4f4e4d",
    height=380,
138                                     width=800, borderwidth=2.4,
139                                     font=("yu gothic ui", 13, "bold"))
140     course_frame.config(highlightbackground="red")
141     course_frame.place(x=100, y=90)
142
143
144
145
146     course_name_label = Label(course_frame, text="Course Name ", bg="white", fg="#4f4e4d",
147                                     font=("yu gothic ui", 13, "bold"))
148     course_name_label.place(x=160, y=65)
149
150     course_name_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#6b6a69
    ",
151                                     font=("yu gothic ui semibold", 12))
152     course_name_entry.place(x=410, y=212, width=335)  # trebuchet ms
153
154     course_name_line = Canvas(root, width=335, height=1.5, bg="#bdb9b1", highlightthickness=0)
155     course_name_line.place(x=410, y=234)
156
157     # ============================================================================
158     # ============================Course Duration=================================
159     # ============================================================================
160
161     course_duration_label = Label(course_frame, text="Course Duration ", bg="white", fg="#
    4f4e4d",
162                                       font=("yu gothic ui", 13, "bold"))
163     course_duration_label.place(x=160, y=115)
164
165     course_duration_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#
    6b6a69",
166                                       font=("yu gothic ui semibold", 12))
167     course_duration_entry.place(x=430, y=262, width=315)  # trebuchet ms
168
169     course_duration_line = Canvas(root, width=315, height=1.5, bg="#bdb9b1",
    highlightthickness=0)
170     course_duration_line.place(x=430, y=284)
171
```

```
172     # ============================================================================
173     # ============================Course Credit=================================
174     # ============================================================================
175 
176     course_credit_label = Label(course_frame, text="Course Credit ", bg="white", fg="#4f4e4d",
177                                         font=("yu gothic ui", 13, "bold"))
178     course_credit_label.place(x=160, y=165)
179 
180     course_credit_entry = Entry(root, highlightthickness=0, relief=FLAT, bg="white", fg="#
    6b6a69",
181                                         font=("yu gothic ui semibold", 12))
182     course_credit_entry.place(x=410, y=312, width=335)  # trebuchet ms
183 
184     course_credit_line = Canvas(root, width=335, height=1.5, bg="#bdb9b1", highlightthickness=
    0)
185     course_credit_line.place(x=410, y=334)
186 
187     reg_date = time.strftime("%Y/%m/%d")
188 
189     # ============================================================================
190     # ============================Register options================================
191     # ============================================================================
192     submit_img = ImageTk.PhotoImage(file='Pics\submit.png')
193 
194     submit = Button(course_frame, image=submit_img,
195                             font=("yu gothic ui", 13, "bold"), relief=FLAT, activebackground="
    white"
196                             , borderwidth=0, background="white", cursor="hand2",command=
    validation)
197     submit.image = submit_img
198     submit.place(x=90, y=267)
199 
200     clear_img = ImageTk.PhotoImage(file='Pics\\clear.png')
201     clear_button = Button(course_frame, image=clear_img,
202                                 font=("yu gothic ui", 13, "bold"), relief=FLAT,
    activebackground="white"
203                                 , borderwidth=0, background="white", cursor="hand2"
204                                 )
205     clear_button.image = clear_img
206     clear_button.place(x=250, y=270)
207 
208     back_img = ImageTk.PhotoImage (file='Pics\\back.png')
209     back_button = Button(course_frame, image=back_img,
210                                 font=("yu gothic ui", 13, "bold"), relief=FLAT,
    activebackground="white"
211                                 , borderwidth=0, background="white", cursor="hand2",command=
    back)
212     back_button.image = back_img
213     back_button.place(x=410, y=270)
214 
215     exit_img = ImageTk.PhotoImage(file='Pics\\exit.png')
216     exit_button = Button(course_frame, image=exit_img,
217                                 font=("yu gothic ui", 13, "bold"), relief=FLAT,
    activebackground="white"
218                                 , borderwidth=0, background="white", cursor="hand2", command=
    exit)
219     exit_button.image = exit_img
220     exit_button.place(x=570, y=270)
221 
222 
223     #root.mainloop()
224 
225 
```

```
226 if __name__=="__main__":
227     course_reg()
```