

```

1 import pickle
2 import random
3 import time as t
4 import datetime
5 import matplotlib.pyplot as g
6 import numpy as np
7 from tkinter import *
8 from PIL import ImageTk,Image
9 from prettytable import PrettyTable
10 from prettytable import DOUBLE_BORDER
11 import mysql.connector as m
12 mycon = m.connect(host = "localhost", user = "root", password = "student", database = "SMS")
13 mycur = mycon.cursor()
14 aPSD = "SMS@sskal" # School Management System.
15
16 def isfloat(num):
17     try:
18         float(num)
19         return True
20     except ValueError:
21         return False
22
23 def captchaGen():
24     letters = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm',
25               'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z']
26     numbers = ['1', '2', '3', '4', '5', '6', '7', '8', '9', '0']
27     symbols = ['~', '!', '@', '$', '%', '^', '&', '*', '?']
28     a = letters[random.randint(0,25)].lower()
29     b = numbers[random.randint(0,9)]
30     c = letters[random.randint(0,25)].upper()
31     d = symbols[random.randint(0,8)]
32     e = numbers[random.randint(0,9)]
33     f = letters[random.randint(0,25)].lower()
34     g = symbols[random.randint(0,8)]
35     h = letters[random.randint(0,25)].upper()
36     captcha = a+b+c+d+e+f+g+h
37     return captcha
38
39 def rectify(N):
40     return 118 + (N-1)*(-3.7)
41
42 #####
43 def tPortal(User):
44     print("~*72)
45     print("~*72)
46     print("=-*14, 'Teacher Portal', "=-*14)
47     print(">>> Welcome",User)
48     print("----")
49     def enroll():
50         print("+*70)
51         print("Instructions: ")
52         print("1. Entering the Enrollment Number is mandatory!.")
53         print("2. Field with unknown value can be skipped by pressing [ENTER].")
54         print("+*70)
55         while True:
56             print("----")
57             print("[Admission Protocol]:")
58             print("+*70)
59             rollno = input("--- Enter the Enrollment Number of Cadet: ")
60             while len(rollno)== 0:
61                 print("--- Entering the Enrollment Number is mandatory.")
62                 rollno = input("    Please Enter the Enrollment Number: ")
63             name = input("--- Enter the Name of the Cadet: ")
64             clss = input("--- Enter the Class the cadet is studying in: ")
65             sect = input("--- Enter the Section the cadet is assigned to: ")
66             phno = input("--- Enter the 10-Digit Phone Number of Cadet: ")
67
68             print("+*70)
69             ACDID = "ACD" + rollno
70             ATDID = "ATD" + rollno
71             record = [rollno,name.title(),clss,sect.upper(),phno,ACDID,ATDID]
72             string = "insert into cData values('{}','{}','{}','{}','{}','{}','{}')"
73             query = (string).format(record[0],record[1],record[2],record[3],record[4],record[5],record[6])
74             mycur.execute(query)
75             mycon.commit()
76             print("Press [ENTER] to continue the Admissions: ")
77             print("--- Else:{Want to Exit}:> Press any key and [Enter]:")
78             prompt = input(">>> ")
79             if len(prompt) != 0:
80                 break
81             print("> Protocol Completed: Data added successfully...")
82
83 def update():
84     while True:
85         print("----")
86         print("[Updation Protocol]:")
87         print("Reference: ")
88         header = ["Rollno", "Name" , "Class", "Section", "Phone number"]
89         table = PrettyTable(header)
90         query = "Select Rollno, Name, Class, Section, Phno from cData"
91         mycur.execute(query)
92         sequence = mycur.fetchall()
93         for i in sequence:
94             table.add_row(i)
95         table.set_style(DOUBLE_BORDER)
96         print(table)
97         print()
98         print("~*63)
99         print("Instructions: ")
100        print("1. Only one field can be updated at a time.")
101        print("2. Struck in middle of updation; then skip all other Entries.")
102        print("3. Basis of updation: ")
103        print("    A. Basis is what you select to update a value.")
104        print("    B. Fields for Basis are always unique to rely on.")
105        rlist = ["Rollno", "Name" , "Class", "Section", "Phone number"]
106        rtable = PrettyTable([1,2,3,4,5])
107        rtable.set_style(DOUBLE_BORDER)
108        rtable.add_row(rlist)

```

```

107 table.add_row((113))
108 print(rtable)
109 fChoice = input("Choose the Index of field you want to update from the above menu: ")
110 print("***63")
111 print("Choose the Basis of Updation: ")
112 print("    1. Rollno")
113 print("    2. Name")
114 bChoice = input(">>> (1 or 2): ")
115 if bChoice == "1":
116     base = "Rollno"
117     bVal = input("Enter the Enrollment number of the cadet: ")
118     if fChoice == "1":
119         field = "Rollno"
120         fVal = input("Enter the new value for field; Rollno: ")
121     elif fChoice == "2":
122         field = "Name"
123         fVal = input("Enter the new value for field; Name: ")
124         fVal = fVal.title()
125     elif fChoice == "3":
126         field = "Class"
127         fVal = input("Enter the new value for field; Class: ")
128     elif fChoice == "4":
129         field = "Section"
130         fVal = input("Enter the new value for field; Section: ")
131         fVal = fVal.upper()
132     elif fChoice == "5":
133         field = "Phno"
134         fVal = input("Enter the new value for field; Phone number: ")
135     else:
136         print("IIError: [Invalid Input for Field.]")
137 elif bChoice == "2":
138     base = "Name"
139     print("Input for this Field must be Accurate!")
140     bVal = input(">>>Enter the Name of the cadet: ")
141     if fChoice == "1":
142         field = "Rollno"
143         fVal = input("Enter the new value for field; Rollno: ")
144     elif fChoice == "2":
145         field = "Name"
146         fVal = input("Enter the new value for field; Name: ")
147         fVal = fVal.title()
148     elif fChoice == "3":
149         field = "Class"
150         fVal = input("Enter the new value for field; Class: ")
151     elif fChoice == "4":
152         field = "Section"
153         fVal = input("Enter the new value for field; Section: ")
154         fVal = fVal.upper()
155     elif fChoice == "5":
156         field = "Phno"
157         fVal = input("Enter the new value for field; Phone number: ")
158     else:
159         print("IIError: [Invalid Input for Field.]")
160 else:
161     print("IIError: [Invalid Input for Basis of Updation.]")
162 string = "Update cData set {} = '{}' where {} = '{}'"
163 query = string.format(field, fVal, base, bVal)
164 mycur.execute(query)
165 mycon.commit()
166 print("***63")
167 print("Value has been updated...")
168 print("---")
169 print("Press [ENTER] to continue the Modification: ")
170 print("--- Else:{Want to Exit}:> Press any key and [Enter]:")
171 prompt = input(">>> ")
172 if len(prompt) != 0:
173     break
174 print("> Protocol Completed: Data Modified successfully...")
175
176 def pAttendance():
177     print("Attendance Portal.")
178     while True:
179         print("+**40")
180         print("1. Mark Attendance.")
181         print("2. Cadet's Attendance History.")
182         print("3. <<< Back")
183         print("+**40")
184         aChoice = input("Enter the index of your choice: ")
185         if aChoice == "1":
186             while True:
187                 print("+**40")
188                 print("1. Overall Attendance.")
189                 print("2. Division wise Attendance.")
190                 print("3. <<< Back")
191                 print("+**40")
192                 mChoice = input("Enter the index of your choice: ")
193                 time = datetime.datetime.now()
194                 tName = "ATD" + str(time.month) + str(time.year)
195                 query = "Show tables"
196                 mycur.execute(query)
197                 data = mycur.fetchall()
198                 tablist = []
199                 for i in data:
200                     tablist.append(i[0])
201
202                 if tName.lower() not in tablist:
203                     string = "Create table if not exists {} select Rollno, Name, Class, Section from cData"
204                     query = string.format(tName)
205                     mycur.execute(query)
206                     mycon.commit()
207                     date = "D" + str(time.day)
208                     try:
209                         string = "Alter table {} add ({} varchar(50))"
210                         query = string.format(tName, date)
211                         mycur.execute(query)
212                         mycon.commit()
213                     except:
214                         def Notice():
215                             print("-----")

```

```

214         print("Attendance for today is already taken.")
215     print("[ Attendance Overwriting Protocol...]")
216     input(">>>")
217
218 def mAttendance():
219     print("Attendance Protocol...")
220     print("    [Fetching table...]")
221     print("    [Fetching Date...]")
222     try:
223         Notice()
224     except:
225         print("", end = "")
226         data = mycur.fetchall()
227         rnos = []
228         names = []
229         for i in data:
230             rnos.append(i[0])
231             names.append(i[1])
232         for i in range(len(rnos)):
233             print("+*40")
234             print("Enrollment number: ", rnos[i])
235             print("--- Name of the Cadet: ", names[i])
236             mark = input("Enter the Attendance [P/A]: ")
237             if mark.islower():
238                 mark = mark.upper()
239             string = "Update {} set {} = '{}' where Rollno = '{}'"
240             query = string.format(tName, date, mark, rnos[i])
241             mycur.execute(query)
242             mycon.commit()
243             print("[", end = "")
244             print(i+1, end = "")
245             print("/", end = "")
246             print(len(rnos), end = "")
247             print("]", end = "")
248             print(" Completed...")
249
250 if mChoice == "1":
251     query = "Select Rollno, Name from {}".format(tName)
252     mycur.execute(query)
253     mAttendance()
254 elif mChoice == "2":
255     print("1. Section A")
256     print("2. Section B")
257     print("3. Section C")
258     Div = input(">>> Enter the Index of the Division: ")
259     if Div == "1":
260         sec = "A"
261     elif Div == "2":
262         sec = "B"
263     elif Div == "3":
264         sec = "C"
265     else:
266         print("IIError: Invaield Input.")
267     if sec in ["A", "B", "C"]:
268         query = "Select Rollno, Name from {} where Section = '{}'".format(tName, sec)
269         mycur.execute(query)
270         mAttendance()
271
272 elif mChoice == "3":
273     break
274 else:
275     print("IIError: Invalid Input")
276
277 elif aChoice == "2":
278     print("Cadet's Attendance History.")
279     print(">>> Here is the Attendance History in (%) of cadets.")
280     query = "Show tables"
281     mycur.execute(query)
282     data = mycur.fetchall()
283     tablist = []
284     for i in data:
285         if i[0][0:3].upper() == "ATD":
286             tablist.append(i[0].lower())
287     query = "Select Rollno, Name, Class, Section from cData"
288     mycur.execute(query)
289     data = mycur.fetchall()
290     header = []
291     for i in tablist:
292         header.append(i[3:])
293     nlist = (["Rollno", "Name", "Class", "Sec"] + header)
294     table = PrettyTable(nlist)
295     table.set_style(DOUBLE_BORDER)
296     for i in data:
297         cData = i
298         plist = []
299         for k in tablist:
300             string = "Select * from {} where Rollno = '{}'"
301             query = string.format(k, i[0])
302             mycur.execute(query)
303             aData = mycur.fetchone()
304             count = 0
305             pcount = 0
306             if aData != None:
307                 for i in range(len(aData)):
308                     if i > 3:
309                         if aData[i].upper() == "P":
310                             pcount += 1
311                             count += 1
312                         percentage = (pcount/count)*100
313                         percentage = round(percentage, 2)
314                         plist.append(percentage)
315             else:
316                 plist.append("AB")
317             tTup = cData + tuple(plist)
318             table.add_row(tTup)
319     print(table)
320
321 elif aChoice == "3":

```

```

322         break
323
324     else:
325         print("IIError: [Invalid Input]")
326
327 def pAcademics():
328     print("~~~~~")
329     print("| Academic Portal. |")
330     print("~~~~~")
331     print(''''-> No Details of Newly admitted Cadets will be updated in Academics field for previous Activities!.'''')
332 def Redirect():
333     print("----")
334     print("Details Entry Protocol...")
335
336     eName = input("Enter the Examination Name: ")
337     tName = Str + eName.upper()
338     query = "Show tables"
339     mycur.execute(query)
340     data = mycur.fetchall()
341     global tablist
342     tablist = []
343     for i in data:
344         tablist.append(i[0].lower())
345     if tName.lower() not in tablist:
346         string = "Create table if not exists {} select Rollno, Name, Class, Section from cData"
347         query = string.format(tName)
348         mycur.execute(query)
349         mycon.commit()
350 def mEntry(Input):
351     sub = input('Enter the Subject Name: ')
352     sub = sub.title()
353     try:
354         query = ("alter table {} add {} varchar(50)".format(tName,sub))
355         mycur.execute(query)
356         mycon.commit()
357     except:
358         def Notice():
359             print("-----")
360             print("Marks Entry for this subject was already done.")
361             print("[ Marks Overwriting Protocol...]")
362             print(">>> To Cancel the protocol, Type: Quit.")
363             print("    --- Else to continue press [Enter]:")
364         print(">>>")
365         print("Marks Entry Protocol:")
366         print("---- [Fetching Resources]",end = "")
367         for i in range(7):
368             t.sleep(1)
369             print(".", end = "")
370         print()
371         try:
372             Notice()
373             qP = input(">>> Give the Confirmation!: ")
374             if qP.upper() == "QUIT":
375                 return
376         except:
377             print("",end = "")
378         rnos = []
379         names = []
380         for i in Input:
381             rnos.append(i[0])
382             names.append(i[1])
383         mM = int(input('Enter the Maximum Marks of the Exam: '))
384         for i in range(len(rnos)):
385             print("+"*40)
386             print('Enrollment number:', rnos[i])
387             print('--- Name of the Cadet:', names[i])
388             oM = int(input('Enter the Marks obtained by Cadet: '))
389             _age = oM*100/mM
390             string = "update {} set {} = '{}{}' where rollno = '{}{}'"
391             query = string.format(tName,sub,_age,rnos[i])
392             mycur.execute(query)
393             mycon.commit()
394             print("[", end = "")
395             print(i+1, end = "")
396             print("/", end = "")
397             print(len(rnos), end = "")
398             print("]", end = "")
399             print(" Completed...")
400 while True:
401     print("+"*40)
402     print("---- 1. Overall Mark_Entry.")
403     print("---- 2. Division wise Mark_Entry.")
404     print("---- 3. <<< Back")
405     print(">>>")
406     eChoice = input("Enter the index of your choice: ")
407     if eChoice == "1":
408         query = "Select Rollno, Name from {}".format(tName)
409         mycur.execute(query)
410         Input = mycur.fetchall()
411         mEntry(Input)
412     elif eChoice == "2":
413         print("- 1. Section A")
414         print("- 2. Section B")
415         print("- 3. Section C")
416         print("----")
417         Div = input(">>> Enter the Index of the Division: ")
418         if Div == "1":
419             sec = "A"
420         elif Div == "2":
421             sec = "B"
422         elif Div == "3":
423             sec = "C"
424         else:
425             print("IIError: Invaild Input.")
426         if sec in ["A","B","C"]:
427             query = "Select Rollno, Name from {} where Section = '{}{}'".format(tName, sec)
428             mycur.execute(query)
429             Input = mycur.fetchall()

```

```

429         mEntry(Input)
430     elif eChoice == "3":
431         break
432     else:
433         print("IIError: Invalid Input")
434 while True:
435     print("+*40)
436     print("--- 1. Mark Entry")
437     print("--- 2. Academic Report.")
438     print("--- 3. <<< Back")
439     print("->")
440     aChoice = input("Enter the index of your choice: ")
441     if aChoice == "1":
442         while True:
443             print("+*40)
444             print("--- 1. Subjective Assessment.")
445             print("--- 2. Internal Assessment.")
446             print("--- 3. Co-Cirricular Activities")
447             print("--- 4. <<< Back")
448             print("+*40)
449             ch = input("Enter the index of your choice: ")
450             if ch == "1":
451                 Str = "SUB"
452                 Redirect()
453             elif ch == "2":
454                 Str = "INT"
455                 Redirect()
456             elif ch == "3":
457                 Str = "CCA"
458                 Redirect()
459             elif ch == "4":
460                 break
461             else:
462                 print("IIError: Invalid Input")
463         elif aChoice == "2":
464             def Marksheet(Div):
465                 print("Cadets' Marksheet")
466                 query = 'show tables'
467                 mycur.execute(query)
468                 data = mycur.fetchall()
469
470                 tablist = []
471                 for i in data:
472                     if i[0][0:3].upper() == Div:
473                         tablist.append(i[0])
474                 query = ("select Rollno,Name,Class,Section from cdata")
475                 mycur.execute(query)
476                 data = mycur.fetchall()
477                 header = []
478                 for i in tablist:
479                     header.append(i[3:].upper())
480                 nlist = ["Rollno","Name","Class","Sec"] + header
481                 table = PrettyTable(nlist)
482                 table.set_style(DOUBLE_BORDER)
483                 for k in data:
484                     cData = k
485                     plist = []
486                     for g in tablist:
487                         query = ('select * from {} where rollno = {}'.format(g,k[0])
488                         mycur.execute(query)
489                         aData = mycur.fetchone()
490                         tCount = 0
491                         Sum = 0
492                         if aData != None:
493                             for i in range(len(aData)):
494                                 if i>3:
495                                     if isfloat(aData[i]):
496                                         Sum += int(eval(aData[i]))
497                                         tCount+=1
498                                     percentage = Sum/tCount
499                                     percentage = round(percentage,2)
500                                     plist.append(percentage)
501                         else:
502                             plist.append('AB')
503                         tTup = cData + tuple(plist)
504                         table.add_row(tTup)
505                 print(table)
506                 # Menu for Selecting Assessment Type!
507                 print("+*40)
508                 print("--- 1. Subjective Assessment.")
509                 print("--- 2. Internal Assessment.")
510                 print("--- 3. Co-Cirricular Activities")
511                 print("->")
512                 ch = input("Enter the Assessment Type to Display Marksheet: ")
513                 print("+*40)
514                 if ch == "1":
515                     Str = "SUB"
516                 elif ch == "2":
517                     Str = "INT"
518                 elif ch == "3":
519                     Str = "CCA"
520                 elif ch == "4":
521                     break
522                 else:
523                     print("IIError: Invalid Input")
524                 if Str in ["SUB","INT","CCA"]:
525                     print("Redirecting", end = "")
526                     for i in range(3):
527                         t.sleep(1)
528                         print(".", end = "")
529                     print()
530                     print()
531                     Marksheet(Str)
532                 if aChoice == "3":
533                     break
534 def Report(Key):
535     global a,b,c,d,e,block,f,g,h,i,Input,j,k,l,m,Rollno
536     Rollno = Key

```

```

536     Remarks = 'REMARKS:  | > 95 : Brilliant | 85 - 95 : Very Good | 75 - 85 : Good | \n 65 - 75 : Satisfactory | < 65 : Improvisation Needed!'
537     print()
538     print(''''The Objective of ICR is to track Student's Progress
539     Individually, in order to help the teachers to be
540     focused towards the Needed.'''')
541     print()
542
543
544     query = "Show Tables"
545     mycur.execute(query)
546     data = mycur.fetchall()
547     tablist = []
548     ATD, SUB, INT, CCA = [],[],[],[]
549     for i in data:
550         tablist.append(i[0])
551     for i in tablist:
552         if i[0:3].upper() == "ATD":
553             ATD.append(i)
554         elif i[0:3].upper() == "SUB":
555             SUB.append(i)
556         elif i[0:3].upper() == "INT":
557             INT.append(i)
558         elif i[0:3].upper() == "CCA":
559             CCA.append(i)
560
561
562     atd_List = []
563     for k in ATD:
564         string = "Select * from {} where Rollno = '{}'"
565         query = string.format(k, Rollno)
566         mycur.execute(query)
567         aData = mycur.fetchone()
568         count = 0
569         pcount = 0
570         if aData != None:
571             for i in range(len(aData)):
572                 if i > 3:
573                     if aData[i].upper() == "P":
574                         pcount += 1
575                         count += 1
576                     percentage = (pcount/count)*100
577                     percentage = round(percentage, 2)
578                     atd_List.append(percentage)
579             else:
580                 atd_List.append(0)
581     if len(atd_List) != 0:
582         p_ATD = str(round(sum(atd_List)/len(atd_List), 2))
583     else:
584         p_ATD = "NIL"
585
586     int_List = []
587     for l in INT:
588         string = "Select * from {} where Rollno = '{}'"
589         query = string.format(l, Rollno)
590         mycur.execute(query)
591         aData = mycur.fetchone()
592         count = 0
593         Sum = 0
594         if aData != None:
595             for i in range(len(aData)):
596                 if i > 3:
597                     if isinstance(aData[i]):
598                         Sum += int(eval(aData[i]))
599                     count += 1
600                     percentage = (Sum/count)
601                     percentage = round(percentage, 2)
602                     int_List.append(percentage)
603             else:
604                 int_List.append(0)
605     if len(int_List) != 0:
606         p_INT = str(round(sum(int_List)/len(int_List), 2))
607     else:
608         p_INT = "NIL"
609
610
611     cca_List = []
612     for m in CCA:
613         string = "Select * from {} where Rollno = '{}'"
614         query = string.format(m, Rollno)
615         mycur.execute(query)
616         aData = mycur.fetchone()
617         count = 0
618         Sum = 0
619         if aData != None:
620             for i in range(len(aData)):
621                 if i > 3:
622                     if isinstance(aData[i]):
623                         Sum += int(eval(aData[i]))
624                     count += 1
625                     percentage = (Sum/count)
626                     percentage = round(percentage, 2)
627                     cca_List.append(percentage)
628             else:
629                 cca_List.append(0)
630     if len(cca_List) != 0:
631         p_CCA = str(round(sum(cca_List)/len(cca_List), 2))
632     else:
633         p_CCA = "NIL"
634
635     query = "select * from cData where Rollno = '{}'.format(Rollno)"
636     mycur.execute(query)
637     data = mycur.fetchall()
638
639
640     Name, Class, Section = data[0][1], data[0][2], data[0][3]
641     Length = rectify(len(Name))
642
643

```

```

643 SUB.sort()
644 sub_List = []
645 header = []
646 for q in SUB:
647     header.append(q[3:].upper())
648 for g in SUB:
649     query = ('select * from {} where rollno = "{}").format(g,Rollno)
650     mycur.execute(query)
651     aData = mycur.fetchone()
652     tCount = 0
653     Sum = 0
654     if aData != None:
655         for i in range(len(aData)):
656             if i>3:
657                 if isfloat(aData[i]):
658                     Sum += int(eval(aData[i]))
659                 tCount+=1
660             percentage = Sum/tCount
661             percentage = round(percentage,2)
662             sub_List.append(percentage)
663     else:
664         sub_List.append(0)
665 if len(sub_List) != 0:
666     p_SUB = str(round(sum(sub_List)/len(sub_List), 2))
667 else:
668     p_SUB = "NIL"
669
670 input(">>> Press ENTER: ")
671 print()
672 print("--- Report Card Generator: Currently Active!")
673 print()
674
675
676
677
678 def verify(cc):
679     global n,j,k,l,m,z,logo,nlogo
680     try:
681         n.destroy()
682     except NameError:
683         print("end = ")
684     cnfcc = Input.get()
685     if cc == cnfcc:
686         Input.delete(0, END)
687         Input.insert(0,"*****")
688         for z in [a,b,c,d,e,block,f,g,h,i,Input,j,k,l,m,o]:
689             z.destroy()
690         Frame = LabelFrame(window, padx = 10, pady = 10, borderwidth = 6)
691         frame = LabelFrame(Frame, padx = 10, pady = 10, borderwidth = 3)
692         Frame.pack()
693         frame.pack()
694         logo = ImageTk.PhotoImage(Image.open(r".\Resources\LogoX.png"))
695         img = Button(frame,image = logo,borderwidth = 0)
696         img.grid(row = 1, column = 1)
697         box = LabelFrame(frame, padx = 10, pady = 10, borderwidth = 0)
698         box.grid(row = 1,column = 2, padx = 10)
699         p = Label(box, text = "SAINIK SCHOOL KALIKIRI", padx = 65)
700         q = Label(box, text = "ANDHRA PRADESH", padx = 65)
701         r = Label(box, text = "sainik.kalikiri@gmail.com", padx = 65)
702         p.pack(),q.pack(),r.pack()
703         nlogo = ImageTk.PhotoImage(Image.open(r".\Resources\nX.png"))
704         Img = Button(frame,image = nlogo,borderwidth = 0)
705         Img.grid(row = 1, column = 3)
706         aFrame = LabelFrame(Frame, padx = 10, pady = 10, borderwidth = 3)
707         aFrame.pack()
708         s = Button(aFrame, text = "Name : {}".format(Name), padx = Length, bg = "AntiqueWhite2")
709         t = Button(aFrame, text = "Roll.no : {}".format(Rollno), padx = 5, bg = "gainsboro")
710         u = Button(aFrame, text = "Class : {}".format(Class), padx = 5, bg = "AntiqueWhite2")
711         v = Button(aFrame, text = "Section : {}".format(Section), padx = 5, bg = "gainsboro")
712         w = Button(aFrame, text = "Attendance Percentage: {}".format(p_ATD), padx = 172, bg = "AntiqueWhite3")
713         s.grid(row = 1, column = 0)
714         t.grid(row = 1, column = 1)
715         u.grid(row = 1, column = 2)
716         v.grid(row = 1, column = 3)
717         w.grid(row = 2, column = 0, columnspan = 4)
718         null0 = Label(Frame, text = "", padx = 95)
719         null0.pack()
720         bFrame = LabelFrame(Frame, padx = 10, pady = 10, borderwidth = 3, bg = "LemonChiffon3")
721         bFrame.pack()
722         Heading01 = Label(bFrame, text = "Subjective Assessments: ", bg = "LemonChiffon3")
723         Heading01.grid(row = 0, column = 0, columnspan = len(header))
724         for text1 in header:
725             z = Button(bFrame, text = text1, padx = 16, borderwidth = "3", bg = "NavajoWhite2")
726             z.grid(row = 1, column = header.index(text1))
727         for text2 in sub_List:
728             z = Button(bFrame, text = str(text2) + "%", padx = 8, borderwidth = "3", bg = "LemonChiffon2")
729             z.grid(row = 2, column = sub_List.index(text2), columnspan = 1)
730         NULL0 = Label(bFrame, text = " ", bg = "LemonChiffon3")
731         NULL0.grid(row = 3, column = 0, columnspan = len(header))
732         OP = Button(bFrame, text = "Overall Percentage: {}".format(p_SUB), borderwidth = 2, bg = "NavajoWhite2")
733         OP.grid(row = 4, column = 0, columnspan = len(header))
734         null1 = Label(Frame, text = "", padx = 95)
735         null1.pack()
736         cFrame = LabelFrame(Frame, padx = 10, pady = 10, borderwidth = 3)
737         cFrame.pack()
738         Internals = Button(cFrame, text = "Internals Score: {}/100".format(p_INT[:2]), borderwidth = 2, padx = 172, bg = "NavajoWhite3")
739         Internals.pack()
740         cca = Button(cFrame, text = "CCA SCORE: {}/100".format(int(p_CCA[:2])), borderwidth = 2, padx = 179, bg = "NavajoWhite3")
741         cca.pack()
742         NULL1 = Label(Frame, text = "", padx = 95)
743         NULL1.pack()
744         Text = Button(Frame, text = Remarks, bg = "misty rose")
745         Text.pack()
746         desg = Label(window, text = "The Principal, Sainik School Kalikiri.", anchor = E)
747         desg.pack()
748
749
750

```

```

751     else:
752         for z in [j,k,l,m]:
753             z.destroy()
754             n = Label(window, text = '''The Captcha Code is Incorrect!
755             Try again: ''', padx = 95)
756             k = Button(window, text = "Verify & Generate", bg = "lightgreen", command = lambda: verify(cc))
757             l = Label(window, text = "", padx = 95)
758             m = Label(window, text = "", padx = 95)
759             for z in [n,k,l,m]:
760                 z.pack()
761
762
763 def Refresh():
764     global cc,h,i,Input,j,k,l,m,n,z
765     try:
766         n.destroy()
767     except NameError:
768         print("",end = "")
769     cc = captchaGen()
770     for z in [h,i,Input,j,k,l,m]:
771         z.destroy()
772     h = Button(window, text = cc, bg = "lightpink")
773     h.pack()
774     i = Label(window, text = "Enter the Captcha Code below:", padx = 95)
775     Input = Entry(window, width = 10, borderwidth = 2)
776     j = Label(window, text = "", padx = 95)
777     k = Button(window, text = "Verify & Generate", bg = "lightgreen", command = lambda: verify(cc))
778     l = Label(window, text = "", padx = 95)
779     m = Label(window, text = "", padx = 95)
780     for z in [h,i,Input,j,k,l,m]:
781         z.pack()
782
783
784
785 window = Tk()
786 window.title("Report Card")
787 window.iconbitmap(r".\Resources\Logo.ico")
788 a = Label(window, text = "", padx = 95)
789 b = Button(window, text = '''SCHOOL MANAGEMENT
790 SYSTEM''', padx = 50, pady = 8, borderwidth = 5, bg = "lightgray")
791 c = Label(window, text = "", padx = 95)
792 d = Label(window, text = "--- REPORT CARD GENERATOR ---", padx = 50)
793 e = Label(window, text = "", padx = 95)
794 Logo = ImageTk.PhotoImage(Image.open(r".\Resources\Logo_.png"))
795 block = Button(image = Logo, borderwidth = 5,bg = "lightyellow")
796 f = Label(window, text = "", padx = 95)
797 cc = captchaGen()
798 g = Label(window, text = "Captcha Code:", padx = 95)
799 h = Button(window, text = cc, bg = "lightpink")
800 i = Label(window, text = "Enter the Captcha Code below:", padx = 95)
801 Input = Entry(window, width = 10, borderwidth = 2)
802 j = Label(window, text = "", padx = 95)
803 k = Button(window, text = "Verify & Generate", bg = "lightgreen", command = lambda: verify(cc))
804
805 l = Label(window, text = "", padx = 95)
806 m = Label(window, text = "", padx = 95)
807 for z in [a,b,c,d,e,block,f,g,h,i,Input,j,k,l,m]:
808     z.pack()
809 o = Button(window,text = "Refresh", bg = "lightblue", command = Refresh)
810 o.place(x = 220, y = 370)
811
812
813 window.mainloop()
814
815 def plotter(Key):
816     query = "show tables"
817     mycur.execute(query)
818     data = mycur.fetchall()
819     tablist = []
820     for i in data:
821         if i[0][0:3].upper() == "SUB":
822             tablist.append(i[0])
823     tablist.sort()
824     xBar = []
825     for i in tablist:
826         xBar.append(i[3:].upper())
827     plist = []
828     for i in tablist:
829         query = "select * from {} where rollno = '{}'".format(i,Key)
830         mycur.execute(query)
831         aData = mycur.fetchone()
832         tCount = 0
833         Sum = 0
834         if aData != None:
835             for i in range(len(aData)):
836                 if i>3:
837                     if isfloat(aData[i]):
838                         Sum += int(eval(aData[i]))
839                     tCount+=1
840                     percentage = Sum/tCount
841                     percentage = round(percentage,2)
842                     plist.append(percentage)
843     else:
844         plist.append(0)
845 g.plot(xBar, plist, color = "black",
846        linestyle = "-.", marker = "o",
847        markeredgcolor = "red",
848        label = "Examwise Marks plot")
849 font = {'family': 'serif',
850        'color': 'darkred',
851        'weight': 'normal',
852        'size': 14,
853        }
854 font_label = {'family': 'serif',
855        'color': 'black',
856        'weight': 'normal',
857        'size': 10,
858        }

```



```

858 g.title('ICPlot - Individual Cadet Plot', fontdict = font)
859 g.xlabel("ASSESSMENTS", fontdict = font_label)
860 g.ylabel("PERCENTAGES", fontdict = font_label)
861 g.legend()
862 g.grid()
863 g.show()
864
865 def remove():
866     verify = input("Enter the Administrative Password: ")
867     while True:
868         if aPSD == verify:
869             header = ["Rollno", "Name", "Class", "Section", "Phone number"]
870             table = PrettyTable(header)
871
872             query = "select Rollno, Name, Class, Section, Phno from cData"
873             mycur.execute(query)
874             data = mycur.fetchall()
875             for i in data:
876                 table.add_row(i)
877             table.set_style(DOUBLE_BORDER)
878             print("Reference: ")
879             print(table)
880             print("+"*60)
881             print("----")
882             rno = input("Enter the the Enrollment number of cadet: ")
883             print("--Do you want to continue the process of Deletion: ")
884             ans = input(">>> press [ENTER]: ")
885             if len(ans) == 0:
886                 fQuery = "Insert into TCLogs Select * from cdata where Rollno = '{}'.format(rno)"
887                 rQuery = "Delete from cdata where Rollno = '{}'.format(rno)"
888                 mycur.execute(fQuery)
889                 mycon.commit()
890                 mycur.execute(rQuery)
891                 mycon.commit()
892                 print("Log Removed..")
893             print("----")
894             print("Press [ENTER] to continue the Removal: ")
895             print("----Else press any key and [Enter]:")
896             prompt = input(">>> ")
897             if len(prompt) != 0:
898                 break
899
900 def pAnnouncements():
901     print("Announcements Portal.")
902     print("----")
903     def mAnnounce():
904         myfile = open(r".\Resources\Announcements.txt", 'a')
905         while True:
906             anc = input("Enter a new Announcement: ")
907             myfile.write("~"+anc)
908             myfile.flush()
909             print("Do you want to continue: ")
910             ans01 = input(">>> Press [y] or [n]: ")
911             if ans01 == "y":
912                 continue
913             else:
914                 break
915     def vAnnounce():
916         myfile = open(r".\Resources\Announcements.txt", 'r')
917         while True:
918             print('Press Enter to view:')
919             input('>>>')
920             i = 1
921             record = myfile.read()
922             data = record.split("~")
923             data.remove(data[0])
924             if len(data) == 0:
925                 print("-> No Announcements yet.")
926             for line in data:
927                 print(i, end = ". ")
928                 print(line)
929                 i += 1
930             print("Press [Enter] to go back!")
931             block = input(">>>")
932             if block == "":
933                 break
934             else:
935                 continue
936     def rAnnounce():
937         print("-----")
938         print("Instruction: ")
939         print("Only one announcement can be removed at a time!.")
940         while True:
941             myfile = open(r".\Resources\Announcements.txt", 'r')
942             i = 1
943             record = myfile.read()
944             data = record.split("~")
945             data.remove(data[0])
946             for line in data:
947                 print(i, end = ". ")
948                 print(line)
949                 i += 1
950             myfile.close()
951             print("----")
952             choice = input("Enter the index of Announcement for removal: ")
953             if choice.isdigit():
954                 cIndex = int(choice)-1
955                 data.remove(data[cIndex])
956                 myfile = open(r".\Resources\Announcements.txt", 'w')
957                 header = "Announcements"
958                 code = ""
959                 for i in data:
960                     code += "~" + i
961                 flowcode = header + code
962                 myfile.write(flowcode)
963                 myfile.flush()
964                 print("Announcement removed!...")
965             else:

```

```

965         print("IIError: [Invalid Input]")
966         print("To continue the process of removal; press [Enter]")
967         ans = input(">>>")
968         if len(ans) == 0:
969             continue
970         else:
971             myfile.close()
972             break
973     while True:
974         print("+*40)
975         print("1. Announce")
976         print("2. View Announcements")
977         print("3. Remove Announcements")
978         print("4. <<< Back")
979         print("+*40)
980         choice = input("Enter the Index of your Choice: ")
981         if choice == '1':
982             mAnnounce()
983         elif choice == '2':
984             vAnnounce()
985         elif choice == '3':
986             rAnnounce()
987         elif choice == '4':
988             break
989         else:
990             print("IIError: Invalid Input")
991
992     def profileSet():
993         print("Loading Account Credentials",end = "")
994         for i in range(3):
995             t.sleep(1)
996             print(".",end = "")
997         print()
998         def Update():
999             print("~~~~~")
1000             print('Protocol: Username Updation')
1001             print("~~~~~")
1002             myfile = open(r'.\Resources\tCredentials.dat','rb')
1003             creds=[]
1004             try:
1005                 while True:
1006                     data = pickle.load(myfile)
1007                     creds.append(data)
1008             except EOFError:
1009                 myfile.close()
1010             usernames = []
1011             passwords = []
1012             for i in creds:
1013                 usernames.append(i[0])
1014                 passwords.append(i[1])
1015             print("+*40)
1016             username = input('Enter your Current Username: ')
1017             nusername = input('Enter your New Username: ')
1018             if username in usernames:
1019                 pos = usernames.index(username)
1020             else:
1021                 print('IUErrror: Invalid Username - Please ReCheck')
1022                 return
1023             while username.lower() == nusername.lower():
1024                 print('Username matches with the previous one!')
1025                 print('- Try changing it into different one')
1026                 print('If you want to retain your previous username: ')
1027                 print('- Cancel the process by Entering "Quit" below. ')
1028                 print("->")
1029                 nusername = input('Enter your New Username: ')
1030             if nusername.upper() == 'QUIT':
1031                 return
1032             print("----")
1033             print('Verification protocol: ')
1034             print('>>> Confirmation to change the Username: ')
1035             vPSD = input('Enter Password: ')
1036             print("----")
1037             cc = captchaGen()
1038             print('Captcha code: ',cc)
1039             cnfcc = input('Enter the 8 character Captcha Code shown above: ')
1040             if cnfcc == cc:
1041                 if vPSD == passwords[pos]:
1042                     print("--- Authentication Successful ---")
1043                     print("Attempting To Change Username", end = "")
1044                     for i in range(5):
1045                         t.sleep(1)
1046                         print(".",end = "")
1047                     print()
1048                     creds[pos][0] = nusername
1049                     myfile=open(r'.\Resources\tCredentials.dat','wb')
1050                     for i in creds:
1051                         pickle.dump(i,myfile)
1052                     myfile.flush()
1053                     print("Username Updated Successfully...")
1054                     myfile.close()
1055                 else:
1056                     print('Access Denied: Unauthorised attempt for changing Username! | Check for the correct Credentials.')
1057             else:
1058                 print('CCErrror: Invalid Input for Captcha Code')
1059     def Change():
1060         print("~~~~~")
1061         print('Protocol: Password Updation')
1062         print("~~~~~")
1063         myfile = open(r'.\Resources\tCredentials.dat','rb')
1064         creds=[]
1065         try:
1066             while True:
1067                 data=pickle.load(myfile)
1068                 creds.append(data)
1069             except EOFError:
1070                 myfile.close()
1071             usernames=[]

```

```

1072 passwords=[]
1073 for i in creds:
1074     usernames.append(i[0])
1075     passwords.append(i[1])
1076 print("+"*40)
1077 username = input('Enter your Username: ')
1078 password = input("Enter Password: ")
1079 if username in usernames:
1080     pos = usernames.index(username)
1081     if passwords[pos] == password:
1082         print("--- Authentication Successful ---")
1083         print("+"*40)
1084         psd = input("Enter your New Password: ")
1085         while password.lower() == psd.lower():
1086             print('Password matches with the previous one!')
1087             print('- Try changing it into different one')
1088             print('If you want to retain your previous Password: ')
1089             print('- Cancel the process by Entering "Quit" below.' )
1090             print(">")
1091             psd = input('Enter your New Password: ')
1092         if psd.upper()=='QUIT':
1093             return
1094         cnfpsd = input("Retype the New Password: ")
1095         if psd == cnfpsd:
1096             print("----")
1097             cc = captchaGen()
1098             print('Captcha code: ',cc)
1099             cnfcc = input('Enter the 8 character Captcha Code shown above: ')
1100             if cnfcc == cc:
1101                 print("Attempting To Change Password", end = "")
1102                 for i in range(5):
1103                     t.sleep(1)
1104                     print(".",end = "")
1105                 print()
1106                 creds[pos][1] = psd
1107                 myfile=open(r'.\Resources\tCredentials.dat','wb')
1108                 for i in creds:
1109                     pickle.dump(i,myfile)
1110                 myfile.flush()
1111                 print(">>> Password Changed Successfully...")
1112                 myfile.close()
1113             else:
1114                 print('CCError: Invalid Input for Captcha Code')
1115                 return
1116         else:
1117             print('IPError: Passwords did not match.')
1118             return
1119     else:
1120         print('Access Denied: Unauthorised attempt for changing Password! | Check for the correct Credentials.')
1121         return
1122 else:
1123     print('IUErrror: Invalid Username - Please ReCheck')
1124     return
1125
1126 def Delete():
1127     print("~~~~~")
1128     print("Protocol: Account Deletion.")
1129     print("~~~~~")
1130     myfile =open(r'.\Resources\tCredentials.dat','rb')
1131     creds=[]
1132     try:
1133         while True:
1134             data=pickle.load(myfile)
1135             creds.append(data)
1136     except EOFError:
1137         myfile.close()
1138     usernames=[]
1139     passwords=[]
1140     for i in creds:
1141         usernames.append(i[0])
1142         passwords.append(i[1])
1143     user =input('Enter your Username for Deletion: ')
1144     psd =input('Enter your Password: ')
1145     if user in usernames:
1146         pos = usernames.index(user)
1147         if passwords[pos] == psd:
1148             print("--- Authentication Successful ---")
1149             print("+"*40)
1150             cc = captchaGen()
1151             print('Captcha code: ',cc)
1152             cnfcc = input('Enter the 8 character Captcha Code shown above: ')
1153             if cnfcc == cc:
1154                 print("Attempting To Delete Account", end = "")
1155                 for i in range(5):
1156                     t.sleep(1)
1157                     print(".",end = "")
1158                 print()
1159                 log = creds[pos]
1160                 creds.remove(log)
1161                 myfile=open(r'.\Resources\tCredentials.dat','wb')
1162                 for i in creds:
1163                     pickle.dump(i,myfile)
1164                 myfile.flush()
1165                 print(">>> Account Deleted Successfully...")
1166                 myfile.close()
1167             else:
1168                 print('CCErrror: Invalid Input for Captcha Code')
1169         else:
1170             print('Access Denied: Unauthorised attempt for Deleting Account! | Check for the correct Credentials.')
1171     else:
1172         print('IUErrror: Invalid Username - Please ReCheck')
1173
1174 while True:
1175     print("+"*40)
1176     print('--- 1.Update username')
1177     print('--- 2.Change password')
1178     print('--- 3.Delete Account')

```

```

1179         print("3. Delete Account",
1180               print('--- 4. Back')
1181               print("->")
1182         psChoice = input("Enter the Index of your Requirement: ")
1183         if psChoice == "1":
1184             Update()
1185         elif psChoice == "2":
1186             Change()
1187         elif psChoice == "3":
1188             Delete()
1189         elif psChoice == "4":
1190             break
1191         else:
1192             print("IIError: Invalid Input")
1193
1194 def Admin():
1195     verify = input("Enter the Administrative Password to continue for Privileged Operations: ")
1196     if verify == aPSD:
1197         while True:
1198             print("="*20)
1199             print(" Menu: ")
1200             print("- 1. User Lookup")
1201             print("- 2. Change Password")
1202             print("- 3. <- Back")
1203             print("->")
1204             oChoice = input(">>> Enter your Choice: ")
1205             if oChoice == "1":
1206
1207                 print("="*20)
1208                 print(" Menu: ")
1209                 print("- 1. Manual Search")
1210                 print("- 2. Auto Search")
1211                 print("- 3. <- Back")
1212                 print("->")
1213                 sChoice = input(">>> Enter your Choice: ")
1214                 if sChoice == "1":
1215                     print("+*40)
1216                     myfile = open(r".\Resources\sCredentials.dat", "rb")
1217                     creds = []
1218                     try:
1219                         while True:
1220                             data = pickle.load(myfile)
1221                             creds.append(data)
1222                     except EOFError:
1223                         myfile.close()
1224                     print("--- Loading Credentials", end = "")
1225                     for i in range(3):
1226                         t.sleep(1)
1227                         print(".", end = "")
1228                     print()
1229                     print("Credentials: ")
1230                     table = PrettyTable(["Username", "Password", "Rollno", "UserType", "Account", "Security"])
1231                     table.set_style(DOUBLE_BORDER)
1232                     for i in creds:
1233                         if len(i) == 2:
1234                             i.append("000")
1235                         if len(i) == 3:
1236                             table.add_row(i+["Standard", "Student", "Binary"])
1237                     print(table)
1238                 elif sChoice == "2":
1239                     myfile = open(r".\Resources\sCredentials.dat", "rb")
1240                     creds = []
1241                     try:
1242                         while True:
1243                             data = pickle.load(myfile)
1244                             creds.append(data)
1245                     except EOFError:
1246                         myfile.close()
1247                     print("--- Loading Credentials", end = "")
1248                     print()
1249                     for i in range(3):
1250                         t.sleep(1)
1251                         print(".", end = "")
1252                     print()
1253                     usr = input("Enter your Username to search for Credentials: ")
1254                     count = 0
1255                     for i in creds:
1256                         if i[0] == usr:
1257                             count = 1
1258                             Record = i
1259                     if count == 0:
1260                         print("Username Not Found!!")
1261                     else:
1262                         table = PrettyTable(["Username", "Password", "Rollno", "UserType", "Account", "Security"])
1263                         table.set_style(DOUBLE_BORDER)
1264                         if len(Record) == 2:
1265                             Record.append("000")
1266                         if len(Record) == 3:
1267                             table.add_row(Record+["Standard", "Student", "Binary"])
1268                         print("Credentials: ")
1269                         print(table)
1270                 elif oChoice == "2":
1271                     myfile = open(r'.\Resources\sCredentials.dat', 'rb')
1272                     creds=[]
1273
1274                     try:
1275                         while True:
1276                             data=pickle.load(myfile)
1277                             creds.append(data)
1278                     except EOFError:
1279                         myfile.close()
1280                     usernames=[]
1281                     passwords=[]
1282                     for i in creds:
1283                         usernames.append(i[0])
1284                         passwords.append(i[1])
1285                     print("+*40)
1286                     username = input('Enter your Username: ')
1287                     if username in usernames:

```

```

1280         username in usernames:
1281             pos = usernames.index(username)
1282             print("--- Intrusion: Authentication Overwrite Successful ---")
1283             print("+"*40)
1284             psd = input("Enter your New Password: ")
1285             cnfpsd = input("Retype the New Password: ")
1286             if psd == cnfpsd:
1287                 print("---")
1288                 cc = captchaGen()
1289                 print('Captcha code: ',cc)
1290                 cnfcc = input('Enter the 8 character Captcha Code shown above: ')
1291                 if cnfcc == cc:
1292                     print("Attempting To Change Password", end = "")
1293                     for i in range(5):
1294                         t.sleep(1)
1295                         print(".",end = "")
1296                     print()
1297                     creds[pos][1] = psd
1298                     myfile=open(r'..\Resources\sCredentials.dat','wb')
1299                     for i in creds:
1300                         pickle.dump(i,myfile)
1301                     myfile.flush()
1302                     print(">>> Password Changed Successfully...")
1303                     myfile.close()
1304                 else:
1305                     print('CCErrror: Invalid Input for Captcha Code')
1306                     return
1307             else:
1308                 print('IPErrror: Passwords did not match.')
1309                 return
1310         elif oChoice == "3":
1311             break
1312     else:
1313         print("===== ! Unauthorised Intrusion Attempted ! =====")
1314
1315 while True:
1316     print("+"*40)
1317     print("--- 1.  Enroll New Cadet")
1318     print("--- 2.  Update Logs")
1319     print("--- 3.  Attendance")
1320     print("--- 4.  Academics")
1321     print("--- 5.  ICReport")
1322     print("--- 6.  Remove Logs")
1323     print("--- 7.  Announcements")
1324     print("--- 8.  Account Settings")
1325     print("--- 9.  Admin Operations")
1326     print("--- 10. LogOut")
1327     print("+"*40)
1328     choice = input("Enter the Index of your Choice: ")
1329     if choice == '1':
1330         enroll()
1331     elif choice == '2':
1332         update()
1333     elif choice == '3':
1334         pAttendance()
1335     elif choice == '4':
1336         pAcademics()
1337     elif choice == '5':
1338         global Rollno,prompt_001,prompt_002,prompt_003
1339         Rollno = input("Enter the Cadet's Enrollment Number: ")
1340         prompt_001 = None
1341         prompt_002 = None
1342         prompt_003 = None
1343         def pPerformance(accessKey):
1344             def try_TITLE():
1345                 global prompt_001,prompt_002,prompt_003
1346                 if prompt_001 == None:
1347                     print("Subjective Assessments: ")
1348                     prompt_001 = "Complete"
1349                 elif prompt_002 == None:
1350                     print("Internal Assessments: ")
1351                     prompt_002 = "Complete"
1352                 elif prompt_003 == None:
1353                     print("CCA Assessments: ")
1354                     prompt_003 = "Complete"
1355
1356         def redirect(i):
1357             print("--- Exam Name: ", end = "")
1358             print(i[3:].upper())
1359             print("->")
1360             header = []
1361             query = "desc {}".format(i)
1362             mycur.execute(query)
1363             data = mycur.fetchall()
1364             for j in data:
1365                 header.append(j[0].title())
1366             table = PrettyTable(header)
1367             table.set_style(DOUBLE_BORDER)
1368             string = "Select * from {} where Rollno = {}".format(i, accessKey)
1369             query = string.format(i, accessKey)
1370             mycur.execute(query)
1371             Record = mycur.fetchall()
1372             for i in Record:
1373                 table.add_row(i)
1374             print(table)
1375
1376         query = "Show tables"
1377         mycur.execute(query)
1378         data = mycur.fetchall()
1379         tablist = []
1380         for i in data:
1381             if i[0][0:3].upper() in ["SUB","INT","CCA"]:
1382                 tablist.append(i[0])
1383         tablist.sort(reverse = True)
1384         print("--- Loading all Marksheets", end = "")
1385         for i in range(3):

```

```

1394         t.sleep(1)
1395         print(".", end = "")
1396     print()
1397     for i in tablist:
1398         if i[0:3].upper() == "SUB":
1399             if prompt_001 != "Complete":
1400                 try_TITLE()
1401                 redirect(i)
1402             elif i[0:3].upper() == "INT":
1403                 if prompt_002 != "Complete":
1404                     try_TITLE()
1405                     redirect(i)
1406             elif i[0:3].upper() == "CCA":
1407                 if prompt_002 != "Complete":
1408                     try_TITLE()
1409                     redirect(i)
1410 pPerformance(Rollno)
1411 while True:
1412     print("=====")
1413     print(" Menu: ")
1414     print("--- 1. Show Development Plots")
1415     print("--- 2. Generate Report Card")
1416     print("--- 3. <- Back")
1417     Ans = input(">>> Enter your Choice: ")
1418     if Ans == "1":
1419         plotter(Rollno)
1420     elif Ans == "2":
1421         Report(Rollno)
1422     elif Ans == "3":
1423         break
1424     else:
1425         print("Invalid Input")
1426
1427 elif choice == '6':
1428     remove()
1429 elif choice == '7':
1430     pAnnouncements()
1431 elif choice == '8':
1432     profileSet()
1433 elif choice == "9":
1434     Admin()
1435 elif choice == '10':
1436     print()
1437     print("-"*15, "LOGGED OUT", "-"*15)
1438     print()
1439     break
1440 else:
1441     print("IIError: Invalid Input")
1442 #####
1443 def sPortal(User, Rollno):
1444     print("_"*72)
1445     print("~"*72)
1446     print("-"*14, 'Student Portal', "-"*14)
1447     print(">>> Welcome", User)
1448     print("----")
1449     accessKey = Rollno
1450     def ACD_Performance():
1451         global prompt_001, prompt_002, prompt_003
1452         prompt_001 = None
1453         prompt_002 = None
1454         prompt_003 = None
1455     def pPerformance(accessKey):
1456         def try_TITLE():
1457             global prompt_001, prompt_002, prompt_003
1458             if prompt_001 == None:
1459                 print("Subjective Assessments: ")
1460                 prompt_001 = "Complete"
1461             elif prompt_002 == None:
1462                 print("Internal Assessments: ")
1463                 prompt_002 = "Complete"
1464             elif prompt_003 == None:
1465                 print("CCA Assessments: ")
1466                 prompt_003 = "Complete"
1467
1468     def redirect(i):
1469         print("--- Exam Name: ", end = "")
1470         print(i[3:].upper())
1471         print("->")
1472         header = []
1473         query = "desc {}".format(i)
1474         mycur.execute(query)
1475         data = mycur.fetchall()
1476         for j in data:
1477             header.append(j[0].title())
1478         table = PrettyTable(header)
1479         table.set_style(DOUBLE_BORDER)
1480         string = "Select * from {} where Rollno = {}".format(i, accessKey)
1481         query = string.format(i, accessKey)
1482         mycur.execute(query)
1483         Record = mycur.fetchall()
1484         for i in Record:
1485             table.add_row(i)
1486         print(table)
1487
1488     query = "Show tables"
1489     mycur.execute(query)
1490     data = mycur.fetchall()
1491     tablist = []
1492     for i in data:
1493         if i[0][0:3].upper() in ["SUB", "INT", "CCA"]:
1494             tablist.append(i[0])
1495     tablist.sort(reverse = True)
1496     print("--- Loading all Marksheets", end = "")
1497     for i in range(3):
1498         t.sleep(1)
1499         print(".", end = "")
1500     print()

```

```

1501         for i in tablist:
1502             if i[0:3].upper() == "SUB":
1503                 if prompt_001 != "Complete":
1504                     try_TITLE()
1505                     redirect(i)
1506             elif i[0:3].upper() == "INT":
1507                 if prompt_002 != "Complete":
1508                     try_TITLE()
1509                     redirect(i)
1510             elif i[0:3].upper() == "CCA":
1511                 if prompt_002 != "Complete":
1512                     try_TITLE()
1513                     redirect(i)
1514
1515         pPerformance(accessKey)
1516
1517     def Report():
1518         global a,b,c,d,e,block,f,g,h,i,Input,j,k,l,m
1519
1520         Remarks = 'REMARKS:  | > 95 : Brilliant | 85 - 95 : Very Good | 75 - 85 : Good | \n 65 - 75 : Satisfactory | < 65 : Improvisation Needed!'
1521         print()
1522         print('''The Objective of ICR is to track Student's Progress
1523 Individually, in order to help the teachers to be
1524 focused towards the Needed.''' )
1525         print()
1526
1527         query = "Show Tables"
1528         mycur.execute(query)
1529         data = mycur.fetchall()
1530         tablist = []
1531         ATD, SUB, INT, CCA = [],[],[],[]
1532         for i in data:
1533             tablist.append(i[0])
1534         for i in tablist:
1535             if i[0:3].upper() == "ATD":
1536                 ATD.append(i)
1537             elif i[0:3].upper() == "SUB":
1538                 SUB.append(i)
1539             elif i[0:3].upper() == "INT":
1540                 INT.append(i)
1541
1542             elif i[0:3].upper() == "CCA":
1543                 CCA.append(i)
1544
1545         atd_List = []
1546         for k in ATD:
1547             string = "Select * from {} where Rollno = '{}'"
1548             query = string.format(k,Rollno)
1549             mycur.execute(query)
1550             aData = mycur.fetchone()
1551             count = 0
1552             pcount = 0
1553             if aData != None:
1554                 for i in range(len(aData)):
1555                     if i > 3:
1556                         if aData[i].upper() == "P":
1557                             pcount += 1
1558                             count += 1
1559                             percentage = (pcount/count)*100
1560                             percentage = round(percentage, 2)
1561                             atd_List.append(percentage)
1562             else:
1563                 atd_List.append(0)
1564         if len(atd_List) != 0:
1565             p_ATD = str(round(sum(atd_List)/len(atd_List), 2))
1566         else:
1567             p_ATD = "NIL"
1568
1569         int_List = []
1570         for l in INT:
1571             string = "Select * from {} where Rollno = '{}'"
1572             query = string.format(l,Rollno)
1573             mycur.execute(query)
1574             aData = mycur.fetchone()
1575             count = 0
1576             Sum = 0
1577             if aData != None:
1578                 for i in range(len(aData)):
1579                     if i > 3:
1580                         if isfloat(aData[i]):
1581                             Sum += int(eval(aData[i]))
1582                             count += 1
1583                             percentage = (Sum/count)
1584                             percentage = round(percentage, 2)
1585                             int_List.append(percentage)
1586             else:
1587                 int_List.append(0)
1588         if len(int_List) != 0:
1589             p_INT = str(round(sum(int_List)/len(int_List), 2))
1590         else:
1591             p_INT = "NIL"
1592
1593
1594         cca_List = []
1595         for m in CCA:
1596             string = "Select * from {} where Rollno = '{}'"
1597             query = string.format(m,Rollno)
1598             mycur.execute(query)
1599             aData = mycur.fetchone()
1600             count = 0
1601             Sum = 0
1602             if aData != None:
1603                 for i in range(len(aData)):
1604                     if i > 3:
1605                         if isfloat(aData[i]):
1606                             Sum += int(eval(aData[i]))
1607                             count += 1

```

```

1608         percentage = (Sum/count)
1609         percentage = round(percentage, 2)
1610         cca_List.append(percentage)
1611     else:
1612         cca_List.append(0)
1613 if len(cca_List) != 0:
1614     p_CCA = str(round(sum(cca_List)/len(cca_List), 2))
1615 else:
1616     p_CCA = "NIL"
1617
1618 query = "select * from cData where Rollno = '{}'.format(Rollno)"
1619 mycur.execute(query)
1620 data = mycur.fetchall()
1621
1622
1623 Name, Class, Section = data[0][1], data[0][2], data[0][3]
1624 Length = rectify(len(Name))
1625
1626
1627 SUB.sort()
1628 sub_List = []
1629 header = []
1630 for q in SUB:
1631     header.append(q[3:].upper())
1632 for g in SUB:
1633     query = ('select * from {} where rollno = "{}").format(g,Rollno)
1634     mycur.execute(query)
1635     aData = mycur.fetchone()
1636     tCount = 0
1637     Sum = 0
1638     if aData != None:
1639         for i in range(len(aData)):
1640             if i>3:
1641                 if isfloat(aData[i]):
1642                     Sum += int(eval(aData[i]))
1643                 tCount+=1
1644             percentage = Sum/tCount
1645             percentage = round(percentage,2)
1646             sub_List.append(percentage)
1647     else:
1648         sub_List.append(0)
1649 if len(sub_List) != 0:
1650     p_SUB = str(round(sum(sub_List)/len(sub_List), 2))
1651 else:
1652     p_SUB = "NIL"
1653 input(">>> Press ENTER: ")
1654 print()
1655 print("--- Report Card Generator: Currently Active!")
1656 print()
1657
1658
1659
1660
1661 def verify(cc):
1662     global n,j,k,l,m,z,logo,nlogo
1663     try:
1664         n.destroy()
1665     except NameError:
1666         print("",end = "")
1667     cnfcc = Input.get()
1668     if cc == cnfcc:
1669         Input.delete(0, END)
1670         Input.insert(0,"*****")
1671         for z in [a,b,c,d,e,block,f,g,h,i,Input,j,k,l,m,o]:
1672             z.destroy()
1673         Frame = LabelFrame(window, padx = 10, pady = 10, borderwidth = 6)
1674         frame = LabelFrame(Frame, padx = 10, pady = 10, borderwidth = 3)
1675
1676         Frame.pack()
1677         frame.pack()
1678         logo = ImageTk.PhotoImage(Image.open(r".\Resources\LogoX.png"))
1679         img = Button(frame,image = logo,borderwidth = 0)
1680         img.grid(row = 1, column = 1)
1681         box = LabelFrame(frame, padx = 10, pady = 10, borderwidth = 0)
1682         box.grid(row = 1,column = 2, padx = 10)
1683         p = Label(box, text = "SAINIK SCHOOL KALIKIRI", padx = 65)
1684         q = Label(box, text = "ANDHRA PRADESH", padx = 65)
1685         r = Label(box, text = "sainik.kalikiri@gmail.com", padx = 65)
1686         p.pack(),q.pack(),r.pack()
1687         nlogo = ImageTk.PhotoImage(Image.open(r".\Resources\nX.png"))
1688         Img = Button(frame,image = nlogo,borderwidth = 0)
1689         Img.grid(row = 1, column = 3)
1690         aFrame = LabelFrame(Frame, padx = 10, pady = 10, borderwidth = 3)
1691         aFrame.pack()
1692         s = Button(aFrame, text = "Name : {}".format(Name), padx = Length, bg = "AntiqueWhite2")
1693         t = Button(aFrame, text = "Roll.no : {}".format(Rollno), padx = 5, bg = "gainsboro")
1694         u = Button(aFrame, text = "Class : {}".format(Class), padx = 5, bg = "AntiqueWhite2")
1695         v = Button(aFrame, text = "Section : {}".format(Section), padx = 5, bg = "gainsboro")
1696         w = Button(aFrame, text = "Attendance Percentage: {}".format(p_ATD), padx = 172, bg = "AntiqueWhite3")
1697         s.grid(row = 1, column = 0)
1698         t.grid(row = 1, column = 1)
1699         u.grid(row = 1, column = 2)
1700         v.grid(row = 1, column = 3)
1701         w.grid(row = 2, column = 0, columnspan = 4)
1702         null0 = Label(Frame, text = "", padx = 95)
1703         null0.pack()
1704         bFrame = LabelFrame(Frame, padx = 10, pady = 10, borderwidth = 3, bg = "LemonChiffon3")
1705         bFrame.pack()
1706         Heading01 = Label(bFrame, text = "Subjective Assessments: ", bg = "LemonChiffon3")
1707         Heading01.grid(row = 0, column = 0, columnspan = len(header))
1708         for text1 in header:
1709             z = Button(bFrame, text = text1, padx = 16, borderwidth = "3", bg = "NavajoWhite2")
1710             z.grid(row = 1, column = header.index(text1))
1711         for text2 in sub_List:
1712             z = Button(bFrame, text = str(text2) + "%", padx = 8, borderwidth = "3", bg = "LemonChiffon2")
1713             z.grid(row = 2, column = sub_List.index(text2), columnspan = 1)
1714         NULL0 = Label(bFrame, text = " ", bg = "LemonChiffon3")
1715         NULL0.grid(row = 3, column = 0, columnspan = len(header))
1716         OP = Button(bFrame, text = "Overall Percentage: {}".format(p_SUB), borderwidth = 2, bg = "NavajoWhite2")

```



```

1710         or = Button(cFrame, text = "Overall Percentage: {}".format(p_OB), borderwidth = 2, bg = "NavajoWhite3",
1711 OP.grid(row = 4, column = 0, columnspan = len(header))
1712 null1 = Label(Frame, text = "", padx = 95)
1713 null1.pack()
1714 cFrame = LabelFrame(Frame, padx = 10, pady = 10, borderwidth = 3)
1715 cFrame.pack()
1716 Internals = Button(cFrame, text = "Internals Score: {}".format(p_INT[:2]), borderwidth = 2, padx = 172, bg = "NavajoWhite3")
1717 Internals.pack()
1718 cca = Button(cFrame, text = "CCA SCORE: {}".format(int(p_CCA[:2])), borderwidth = 2, padx = 179, bg = "NavajoWhite3")
1719 cca.pack()
1720 NULL1 = Label(Frame, text = "", padx = 95)
1721 NULL1.pack()
1722 Text = Button(Frame, text = Remarks, bg = "misty rose")
1723 Text.pack()
1724 desg = Label(window, text = "The Principal, Sainik School Kalikiri.", anchor = E)
1725 desg.pack()
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736     else:
1737         for z in [j,k,l,m]:
1738             z.destroy()
1739             n = Label(window, text = "'The Captcha Code is Incorrect!'
1740             Try again: '", padx = 95)
1741             k = Button(window, text = "Verify & Generate", bg = "lightgreen", command = lambda: verify(cc))
1742             l = Label(window, text = "", padx = 95)
1743             m = Label(window, text = "", padx = 95)
1744             for z in [n,k,l,m]:
1745                 z.pack()
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
def Refresh():
    global cc,h,i,Input,j,k,l,m,n,z
    try:
        n.destroy()
    except NameError:
        print("",end = "")
    cc = captchaGen()
    for z in [h,i,Input,j,k,l,m]:
        z.destroy()
    h = Button(window, text = cc, bg = "lightpink")
    h.pack()
    i = Label(window, text = "Enter the Captcha Code below:", padx = 95)
    Input = Entry(window, width = 10, borderwidth = 2)
    j = Label(window, text = "", padx = 95)
    k = Button(window, text = "Verify & Generate", bg = "lightgreen", command = lambda: verify(cc))
    l = Label(window, text = "", padx = 95)
    m = Label(window, text = "", padx = 95)
    for z in [h,i,Input,j,k,l,m]:
        z.pack()

window = Tk()
window.title("Report Card")
window.iconbitmap(r".\Resources\Logo.ico")
a = Label(window, text = "", padx = 95)
b = Button(window, text = "'SCHOOL MANAGEMENT
SYSTEM'", padx = 50, pady = 8, borderwidth = 5, bg = "lightgray")
c = Label(window, text = "", padx = 95)
d = Label(window, text = "--- REPORT CARD GENERATOR ---", padx = 50)
e = Label(window, text = "", padx = 95)
Logo = ImageTk.PhotoImage(Image.open(r".\Resources\Logo.png"))
block = Button(image = Logo, borderwidth = 5,bg = "lightyellow")
f = Label(window, text = "", padx = 95)
cc = captchaGen()
g = Label(window, text = "Captcha Code:", padx = 95)
h = Button(window, text = cc, bg = "lightpink")
i = Label(window, text = "Enter the Captcha Code below:", padx = 95)
Input = Entry(window, width = 10, borderwidth = 2)
j = Label(window, text = "", padx = 95)
k = Button(window, text = "Verify & Generate", bg = "lightgreen", command = lambda: verify(cc))
l = Label(window, text = "", padx = 95)
m = Label(window, text = "", padx = 95)
for z in [a,b,c,d,e,block,f,g,h,i,Input,j,k,l,m]:
    z.pack()
o = Button(window,text = "Refresh", bg = "lightblue", command = Refresh)
o.place(x = 220, y = 370)

window.mainloop()

def pAnnouncements():
    myfile = open(r".\Resources\Announcements.txt",'r')
    while True:
        print('Press Enter to view:')
        input('>>>')
        i = 1
        record = myfile.read()
        data = record.split("\n")
        data.remove(data[0])
        if len(data) == 0:
            print(">-> No Announcements Yet!")
        for line in data:
            print(i, end = ". ")
            print(line)
            i += 1
        print("Press [Enter] to go back!")
        block = input(">>>")
        if block == "":
            break
        else:
            continue

def profileSet():
    print("Loading Account Credentials",end = "")
    for i in range(3):
        + sleep(1)

```

```

1822         t.sleep(1)
1823         print(".",end = "")
1824     print()
1825     def Update():
1826         print("~~~~~")
1827         print('Protocol: Username Updation')
1828         print("~~~~~")
1829         myfile = open(r'.\Resources\sCredentials.dat','rb')
1830         creds =[]
1831         try:
1832             while True:
1833                 data = pickle.load(myfile)
1834                 creds.append(data)
1835         except EOFError:
1836             myfile.close()
1837         usernames = []
1838         passwords = []
1839         for i in creds:
1840             usernames.append(i[0])
1841             passwords.append(i[1])
1842         print("+*40)
1843         username = input('Enter your Current Username: ')
1844         nusername = input('Enter your New Username: ')
1845         if username in usernames:
1846             pos = usernames.index(username)
1847         else:
1848             print('IUErrror: Invalid Username - Please ReCheck')
1849             return
1850         while username.lower() == nusername.lower():
1851             print('Username matches with the previous one!')
1852             print('- Try changing it into different one')
1853             print('If you want to retain your previous username: ')
1854             print('- Cancel the process by Entering "Quit" below. ')
1855             print("->")
1856             nusername = input('Enter your New Username: ')
1857         if nusername.upper() == 'QUIT':
1858             return
1859         print("----")
1860         print('Verification protocol: ')
1861         print('>>> Confirmation to change the Username: ')
1862         vPSD = input(r'Enter Password: ')
1863         print("----")
1864         cc = captchaGen()
1865         print('Captcha code: ',cc)
1866         cnfcc = input('Enter the 8 character Captcha Code shown above: ')
1867         if cnfcc == cc:
1868             if vPSD == passwords[pos]:
1869                 print("---- Authentication Successful ----")
1870                 print("Attempting To Change Username", end = "")
1871                 for i in range(5):
1872                     t.sleep(1)
1873                     print(".",end = "")
1874                 print()
1875                 creds[pos][0] = nusername
1876
1877                 myfile=open(r'.\Resources\sCredentials.dat','wb')
1878                 for i in creds:
1879                     pickle.dump(i,myfile)
1880                 myfile.flush()
1881                 print("Username Updated Successfully...")
1882                 myfile.close()
1883             else:
1884                 print('Access Denied: Unauthorised attempt for changing Username! | Check for the correct Credentials.')
1885         else:
1886             print('CCError: Invalid Input for Captcha Code')
1887     def Change():
1888         print("~~~~~")
1889         print('Protocol: Password Updation')
1890         print("~~~~~")
1891         myfile = open(r'.\Resources\sCredentials.dat','rb')
1892         creds=[]
1893         try:
1894             while True:
1895                 data=pickle.load(myfile)
1896                 creds.append(data)
1897         except EOFError:
1898             myfile.close()
1899         usernames=[]
1900         passwords=[]
1901         for i in creds:
1902             usernames.append(i[0])
1903             passwords.append(i[1])
1904         print("+*40)
1905         username = input('Enter your Username: ')
1906         password = input("Enter Password: ")
1907         if username in usernames:
1908             pos = usernames.index(username)
1909             if passwords[pos] == password:
1910                 print("---- Authentication Successful ----")
1911                 print("+*40)
1912                 psd = input("Enter your New Password: ")
1913                 while password.lower() == psd.lower():
1914                     print('Password matches with the previous one!')
1915                     print('- Try changing it into different one')
1916                     print('If you want to retain your previous Password: ')
1917                     print('- Cancel the process by Entering "Quit" below. ')
1918                     print("->")
1919                     psd = input('Enter your New Password: ')
1920                 if psd.upper() == 'QUIT':
1921                     return
1922                 cnfpsd = input("Retype the New Password: ")
1923                 if psd == cnfpsd:
1924                     print("----")
1925                     cc = captchaGen()
1926                     print('Captcha code: ',cc)
1927                     cnfcc = input('Enter the 8 character Captcha Code shown above: ')
1928                     if cnfcc == cc:
1929                         print("Attempting To Change Password", end = "")
1930                         for i in range(5):

```

```

1930         t.sleep(1)
1931         print(".",end = "")
1932         print()
1933         creds[pos][1] = psd
1934         myfile=open(r'..\Resources\sCredentials.dat','wb')
1935         for i in creds:
1936             pickle.dump(i,myfile)
1937         myfile.flush()
1938         print(">>> Password Changed Successfully...")
1939         myfile.close()
1940     else:
1941         print('CCErrror: Invalid Input for Captcha Code')
1942         return
1943
1944     else:
1945         print('IPError: Passwords did not match.')
1946         return
1947
1948     else:
1949         print('Access Denied: Unauthorised attempt for changing Password! | Check for the correct Credentials.')
1950         return
1951
1952     else:
1953         print('IUError: Invalid Username - Please ReCheck')
1954         return
1955
1956 def Delete():
1957     print("~~~~~")
1958     print("Protocol: Account Deletion.")
1959     print("~~~~~")
1960     myfile =open(r'..\Resources\sCredentials.dat','rb')
1961     creds=[]
1962     try:
1963         while True:
1964             data=pickle.load(myfile)
1965             creds.append(data)
1966     except EOFError:
1967         myfile.close()
1968     usernames=[]
1969     passwords=[]
1970     for i in creds:
1971         usernames.append(i[0])
1972         passwords.append(i[1])
1973     user =input('Enter your Username for Deletion: ')
1974     psd =input('Enter your Password: ')
1975     if user in usernames:
1976         pos = usernames.index(user)
1977         if passwords[pos] == psd:
1978             print("--- Authentication Successful ---")
1979             print("+"*40)
1980             cc = captchaGen()
1981             print('Captcha code: ',cc)
1982             cnfcc = input('Enter the 8 character Captcha Code shown above: ')
1983             if cnfcc == cc:
1984                 print("Attempting To Delete Account", end = "")
1985                 for i in range(5):
1986                     t.sleep(1)
1987                     print(".",end = "")
1988                 print()
1989                 log = creds[pos]
1990                 creds.remove(log)
1991                 myfile=open(r'..\Resources\sCredentials.dat','wb')
1992                 for i in creds:
1993                     pickle.dump(i,myfile)
1994                 myfile.flush()
1995                 print(">>> Account Deleted Successfully...")
1996                 myfile.close()
1997             else:
1998                 print('CCErrror: Invalid Input for Captcha Code')
1999
2000         else:
2001             print('Access Denied: Unauthorised attempt for Deleting Account! | Check for the correct Credentials.')
2002
2003     else:
2004         print('IUError: Invalid Username - Please ReCheck')
2005
2006 while True:
2007     print("+"*40)
2008     print('--- 1.Update username')
2009     print('--- 2.Change password')
2010     print('--- 3.Delete Account')
2011     print('--- 4. Back')
2012     print("->")
2013     psChoice = input("Enter the Index of your Requirement: ")
2014     if psChoice == "1":
2015         Update()
2016     elif psChoice == "2":
2017         Change()
2018     elif psChoice == "3":
2019         Delete()
2020     elif psChoice == "4":
2021         break
2022     else:
2023         print("IIError: Invalid Input")
2024
2025 while True:
2026     print("+"*40)
2027     print("1. Academic Performance")
2028     print("2. Report Card")
2029     print("3. Announcements")
2030     print("4. Account Settings")
2031     print("5. Logout")
2032     print("+"*40)
2033     choice = input("Enter the Index of your Choice: ")
2034     if choice == '1':
2035         ACD_Performance()
2036     elif choice == '2':
2037         Report()
2038     elif choice == '3':
2039         pAnnouncements()
2040     elif choice == '4':
2041         profileSet()
2042     elif choice == '5':

```

[illegible]

```

2144         break
2145     else:
2146         print("Invalid Input")
2147 elif Login_mode01 == '2':
2148     while True:
2149         print("+*40)
2150         print('--- 1. Login')
2151         print('--- 2. Register')
2152         print("--- 3. <- Back")
2153         print("->")
2154         Login_mode02 = input("Enter your choice (1 or 2): ")
2155         myfile = open(r".\Resources\sCredentials.dat", 'rb')
2156         creds = []
2157         try:
2158             while True:
2159                 data = pickle.load(myfile)
2160                 creds.append(data)
2161         except EOFError:
2162             myfile.close()
2163         usernames = []
2164         passwords = []
2165         for i in creds:
2166             usernames.append(i[0])
2167             passwords.append(i[1])
2168         if Login_mode02 == '1':
2169             print("+*40)
2170             usr = input("Enter Username: ")
2171             psd = input("Enter Password: ")
2172             if usr in usernames:
2173                 pos = usernames.index(usr)
2174                 if passwords[pos] == psd:
2175                     try:
2176                         Rno = creds[pos][2]
2177                     except IndexError:
2178                         Rno = ""
2179                     print(">>> Authentication Successful <<<")
2180                     print("--- Loading Portal", end = "")
2181                     for i in range(5):
2182                         print(".",end = "")
2183                         t.sleep(1)
2184                     print()
2185                     sPortal(usr,Rno)
2186                 else:
2187                     print('IPError: Invalid Password entered.')
2188             else:
2189                 print('IUError: Invalid Username - Please ReCheck')
2190         elif Login_mode02 == '2':
2191             myfile = open(r".\Resources\sCredentials.dat", 'ab')
2192             print("+*40)
2193             verify = input("Enter your Roll number: ")
2194             query = ("select Rollno from cDATA")
2195             mycur.execute(query)
2196             data = mycur.fetchall()
2197             record = []
2198             for i in data:
2199                 record.append(i[0])
2200             print("----")
2201             if verify in record:
2202                 print("+*40)
2203                 usr = input('Enter a Username: ')
2204                 if usr not in usernames:
2205                     psd = input('Enter the Password: ')
2206                     cnfpsd = input('Confirm your Password: ')
2207                     print("+*40)
2208                     cc = captchaGen()
2209                     print("Captcha Code:" , end = " ")
2210                     print(cc)
2211                     cnfcc = input('Enter the 8 character Captcha Code shown above: ')
2212                     print("+*40)
2213                     if cc == cnfcc:
2214                         if psd == cnfpsd:
2215                             List = [usr,psd,verify]
2216                             pickle.dump(List, myfile)
2217                             myfile.flush()
2218                             print("--- Adding User", end = "")
2219                             for i in range(5):
2220                                 print(".",end = "")
2221                                 t.sleep(1)
2222                             print()
2223                             print('User added successfully.')
2224                             print('- Please Re-login to continue...')
2225                         else:
2226                             print('IPError: Passwords did not match.')
2227                     else:
2228                         print('CCError: Invalid Input for Captcha Code')
2229                 else:
2230                     print('User with this username already exists: Please try again...')
2231             else:
2232                 print('The cadet with this Roll number does not Exist.')
2233         elif Login_mode02 == '3':
2234             break
2235         else:
2236             print("Invalid Input")
2237
2238     elif Login_mode01 == "3":
2239         break
2240     else:
2241         print("IIError: Invalid Input")
2242
2243 elif Log == "2":
2244     quit()
2245 else:
2246     print("IIError: Invalid Input")

```