SOURCE CODE

```
import pickle
import random
import time as t
import datetime
import matplotlib.pyplot as g
import numpy as np
from tkinter import *
from PIL import ImageTk,Image
from prettytable import PrettyTable
from prettytable import DOUBLE_BORDER
import mysql.connector as m
mycon = m.connect(host = "localhost", user = "root", password = "student", database =
"SMS")
mycur = mycon.cursor()
aPSD = "SMS@sskal" # School Management System.

def isfloat(num):
     try:
       float(num)
       return True
     except ValueError:
       return False

def captchaGen():
     letters = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm',
                'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z']
     numbers = ['1', '2', '3', '4', '5', '6', '7', '8', '9', '0']
     symbols = ['~', '!', '@', '$', '%', '^', '&', '*', '?']
     a = letters[random.randint(0,25)].lower()
     b = numbers[random.randint(0,9)]
     c = letters[random.randint(0,25)].upper()
     d = symbols[random.randint(0,8)]
     e = numbers[random.randint(0,9)]
     f = letters[random.randint(0,25)].lower()
     g = symbols[random.randint(0,8)]
     h = letters[random.randint(0,25)].upper()
     captcha = a+b+c+d+e+f+g+h
     return captcha

def rectify(N):
    return 118 + (N-1)*(-3.7)

##########################################################################################
############
def tPortal(User):
     print("_"*72)
     print("~"*72)
     print("=-"*14, 'Teacher Portal', "-="*14)
     print(">>> Welcome",User)
     print("---")
     def enroll():
          print("+"*70)
          print("Instructions: ")
          print("1. Entering the Enrollment Number is mandatory!.")
          print("2. Field with unknown value can be skipped by pressing [ENTER].")
          print("+"*70)
          while True:
               print("---")
               print("[Admission Protocol]:")
               print("+"*70)
               rollno = input("--- Enter the Enrollment Number of Cadet: ")
               while len(rollno)== 0:
                    print("--- Entering the Enrollment Number is mandatory.")
```

```
                    rollno = input("    Please Enter the Enrollment Number: ")
               name = input("--- Enter the Name of the Cadet: ")
               clss = input("--- Enter the Class the cadet is studying in: ")
               sect = input("--- Enter the Section the cadet is assigned to: ")
               phno = input("--- Enter the 10-Digit Phone Number of Cadet: ")
               print("+"*70)
               ACDID = "ACD" + rollno
               ATDID = "ATD" + rollno
               record = [rollno,name.title(),clss,sect.upper(),phno,ACDID,ATDID]
               string = "insert into cData values('{}','{}','{}','{}','{}','{}','{}')"
               query =
(string).format(record[0],record[1],record[2],record[3],record[4],record[5],record[6])
               mycur.execute(query)
               mycon.commit()
               print("Press [ENTER] to continue the Admissions: ")
               print("--- Else:{Want to Exit}:-> Press any key and [Enter]:")
               prompt = input(">>> ")
               if len(prompt) != 0:
                    break
          print("-> Protocol Completed: Data added successfully...")

     def update():
          while True:
               print("---")
               print("[Updation Protocol]:")
               print("Reference: ")
               header = ["Rollno", "Name" , "Class", "Section", "Phone number"]
               table = PrettyTable(header)
               query = "Select Rollno, Name, Class, Section, Phno from cData"
               mycur.execute(query)
               sequence = mycur.fetchall()
               for i in sequence:
                    table.add_row(i)
               table.set_style(DOUBLE_BORDER)
               print(table)
               print()
               print("*"*63)
               print("Instructions: ")
               print("1. Only one field can be updated at a time.")
               print("2. Struck in middle of updation; then skip all other Entries.")
               print("3. Basis of updation: ")
               print("    A. Basis is what you select to update a value.")
               print("    B. Fields for Basis are always unique to rely on.")
               rlist = ["Rollno", "Name" , "Class", "Section", "Phone number"]
               rtable = PrettyTable([1,2,3,4,5])
               rtable.set_style(DOUBLE_BORDER)
               rtable.add_row(rlist)
               print(rtable)
               fChoice = input("Choose the Index of field you want to update from the
above menu: ")
               print("*"*63)
               print("Choose the Basis of Updation: ")
               print("    1. Rollno")
               print("    2. Name")
               bChoice = input(">>> (1 or 2): ")
               if bChoice == "1":
                    base = "Rollno"
                    bVal = input("Enter the Enrollment number of the cadet: ")
                    if fChoice == "1":
                         field = "Rollno"
                         fVal = input("Enter the new value for field; Rollno: ")
                    elif fChoice == "2":
                         field = "Name"
                         fVal = input("Enter the new value for field; Name: ")
```

```
                    fVal = fVal.title()
               elif fChoice == "3":
                    field = "Class"
                    fVal = input("Enter the new value for field; Class: ")
               elif fChoice == "4":
                    field = "Section"
                    fVal = input("Enter the new value for field; Section: ")
                    fVal = fVal.upper()
               elif fChoice == "5":
                    field = "Phno"
                    fVal = input("Enter the new value for field; Phone number: ")
               else:
                    print("IIError: [Invalid Input for Field.]")
          elif bChoice == "2":
               base = "Name"
               print("Input for this Field must be Accurate!")
               bVal = input(">>>Enter the Name of the cadet: ")
               if fChoice == "1":
                    field = "Rollno"
                    fVal = input("Enter the new value for field; Rollno: ")
               elif fChoice == "2":
                    field = "Name"
                    fVal = input("Enter the new value for field; Name: ")
                    fVal = fVal.title()
               elif fChoice == "3":
                    field = "Class"
                    fVal = input("Enter the new value for field; Class: ")
               elif fChoice == "4":
                    field = "Section"
                    fVal = input("Enter the new value for field; Section: ")
                    fVal = fVal.upper()
               elif fChoice == "5":
                    field = "Phno"
                    fVal = input("Enter the new value for field; Phone number: ")
               else:
                    print("IIError: [Invalid Input for Field.]")
          else:
               print("IIError: [Invalid Input for Basis of Updation.]")
          string = "Update cData set {} = '{}' where {} = '{}'"
          query = string.format(field,fVal,base,bVal)
          mycur.execute(query)
          mycon.commit()
          print("*"*63)
          print("Value has been updated...")
          print("---")
          print("Press [ENTER] to continue the Modification: ")
          print("--- Else:{Want to Exit}:-> Press any key and [Enter]:")
          prompt = input(">>> ")
          if len(prompt) != 0:
               break
     print("-> Protocol Completed: Data Modified successfully...")

def pAttendance():
     print("Attendance Portal.")
     while True:
          print("+"*40)
          print("1. Mark Attendance.")
          print("2. Cadet's Attendance History.")
          print("3. <<< Back")
          print("+"*40)
          aChoice = input("Enter the index of your choice: ")
          if aChoice == "1":
               while True:
                    print("+"*40)
```

```
                    print("1. Overall Attendance.")
                    print("2. Division wise Attendance.")
                    print("3. <<< Back")
                    print("+"*40)
                    mChoice = input("Enter the index of your choice: ")
                    time = datetime.datetime.now()
                    tName = "ATD" + str(time.month) + str(time.year)
                    query = "Show tables"
                    mycur.execute(query)
                    data = mycur.fetchall()
                    tablist = []
                    for i in data:
                         tablist.append(i[0])
                    if tName.lower() not in tablist:
                         string = "Create table if not exists {} select Rollno,
Name, Class, Section from cData"
                         query = string.format(tName)
                         mycur.execute(query)
                         mycon.commit()
                    date = "D" + str(time.day)
                    try:
                         string = "Alter table {} add ({} varchar(50))"
                         query = string.format(tName,date)
                         mycur.execute(query)
                         mycon.commit()
                    except:
                         def Notice():
                              print("---------------------------------------")
                              print("Attendance for today is already taken.")
                              print("[ Attendance Overwriting Protocol...]")
                              input(">>>")
                    def mAttendance():
                         print("Attendance Protocol...")
                         print("    [Fetching table...]")
                         print("    [Fetching Date...]")
                         try:
                              Notice()
                         except:
                              print("",end = "")
                         data = mycur.fetchall()
                         rnos = []
                         names = []
                         for i in data:
                              rnos.append(i[0])
                              names.append(i[1])
                         for i in range(len(rnos)):
                              print("+"*40)
                              print("Enrollment number: ",rnos[i])
                              print("--- Name of the Cadet: ",names[i])
                              mark = input("Enter the Attendance [P/A]: ")
                              if mark.islower():
                                   mark = mark.upper()
                              string = "Update {} set {} = '{}' where Rollno =
'{}'"
                              query = string.format(tName, date, mark, rnos[i])
                              mycur.execute(query)
                              mycon.commit()
                              print("[", end = "")
                              print(i+1, end = "")
                              print("/", end = "")
                              print(len(rnos), end = "")
                              print("]", end = "")
                              print(" Completed...")
```

```
                    if mChoice == "1":
                         query = "Select Rollno, Name from {}".format(tName)
                         mycur.execute(query)
                         mAttendance()
                    elif mChoice == "2":
                         print("1. Section A")
                         print("2. Section B")
                         print("3. Section C")
                         Div = input(">>> Enter the Index of the Division: ")
                         if Div == "1":
                              sec = "A"
                         elif Div == "2":
                              sec = "B"
                         elif Div == "3":
                              sec = "C"
                         else:
                              print("IIError: Invaild Input.")
                         if sec in ["A","B","C"]:
                              query = "Select Rollno, Name from {} where Section
= '{}'".format(tName, sec)
                              mycur.execute(query)
                              mAttendance()

                    elif mChoice == "3":
                         break
                    else:
                         print("IIError: Invalid Input")

               elif aChoice == "2":
                    print("Cadet's Attendance History.")
                    print(">>> Here is the Attendance History in (%) of cadets.")
                    query = "Show tables"
                    mycur.execute(query)
                    data = mycur.fetchall()
                    tablist = []
                    for i in data:
                         if i[0][0:3].upper() == "ATD":
                              tablist.append(i[0].lower())
                    query = "Select Rollno, Name, Class, Section from cData"
                    mycur.execute(query)
                    data = mycur.fetchall()
                    header = []
                    for i in tablist:
                         header.append(i[3:])
                    nlist = (["Rollno","Name","Class","Sec"]+header)
                    table = PrettyTable(nlist)
                    table.set_style(DOUBLE_BORDER)
                    for i in data:
                         cData = i
                         plist = []
                         for k in tablist:
                              string = "Select * from {} where Rollno = '{}'"
                              query = string.format(k,i[0])
                              mycur.execute(query)
                              aData = mycur.fetchone()
                              count = 0
                              pcount = 0
                              if aData != None:
                                   for i in range(len(aData)):
                                        if i > 3:
                                             if aData[i].upper() == "P":
                                                  pcount += 1
                                             count += 1
                                   percentage = (pcount/count)*100
```

```
                                    percentage = round(percentage, 2)
                                    plist.append(percentage)
                               else:
                                    plist.append("AB")
                          tTup = cData + tuple(plist)
                          table.add_row(tTup)
                     print(table)

               elif aChoice == "3":
                    break

               else:
                    print("IIError: [Ivalid Input]")

     def pAcademics():
          print("~~~~~~~~~~~~~~~~~~~~")
          print("| Academic Portal. |")
          print("~~~~~~~~~~~~~~~~~~~~")
          print('''-> No Details of Newly admitted Cadets will be updated in Academics
field for previous Activities!.''')
          def Redirect():
             print("---")
             print("Details Entry Protocol...")
             eName = input("Enter the Examination Name: ")
             tName = Str + eName.upper()
             query = "Show tables"
             mycur.execute(query)
             data = mycur.fetchall()
             global tablist
             tablist = []
             for i in data:
                 tablist.append(i[0].lower())
             if tName.lower() not in tablist:
                 string = "Create table if not exists {} select Rollno, Name, Class,
Section from cData"
                 query = string.format(tName)
                 mycur.execute(query)
                 mycon.commit()
             def mEntry(Input):
                 sub = input('Enter the Subject Name: ')
                 sub = sub.title()
                 try:
                     query = ("alter table {} add {} varchar(50)").format(tName,sub)
                     mycur.execute(query)
                     mycon.commit()
                 except:
                     def Notice():
                         print("-----------------------------------------------")
                         print("Marks Entry for this subject was already done.")
                         print("[ Marks Overwriting Protocol...]")
                         print(">>> To Cancel the protocol, Type: Quit.")
                         print("    --- Else to continue press [Enter]:")
                 print("->")
                 print("Marks Entry Protocol:")
                 print("--- [Fetching Resources]",end = "")
                 for i in range(7):
                     t.sleep(1)
                     print(".", end = "")
                 print()
                 try:
                     Notice()
                     qP = input(">>> Give the Confirmation!: ")
                     if qP.upper() == "QUIT":
                         return
```

```
                except:
                    print("",end = "")
                rnos = []
                names = []
                for i in Input:
                    rnos.append(i[0])
                    names.append(i[1])
                mM = int(input('Enter the Maximum Marks of the Exam: '))
                for i in range(len(rnos)):
                    print("+"*40)
                    print('Enrollment number:',rnos[i])
                    print('--- Name of the Cadet:',names[i])
                    oM = int(input('Enter the Marks obtained by Cadet: '))
                    _age = oM*100/mM
                    string = "update {} set {} ='{}' where rollno = '{}'"
                    query = string.format(tName,sub,_age,rnos[i])
                    mycur.execute(query)
                    mycon.commit()
                    print("[", end = "")
                    print(i+1, end = "")
                    print("/", end = "")
                    print(len(rnos), end = "")
                    print("]", end = "")
                    print(" Completed...")
            while True:
                print("+"*40)
                print("--- 1. Overall Mark_Entry.")
                print("--- 2. Division wise Mark_Entry.")
                print("--- 3. <<< Back")
                print("->")
                eChoice = input("Enter the index of your choice: ")
                if eChoice == "1":
                    query = "Select Rollno, Name from {}".format(tName)
                    mycur.execute(query)
                    Input = mycur.fetchall()
                    mEntry(Input)
                elif eChoice == "2":
                    print("- 1. Section A")
                    print("- 2. Section B")
                    print("- 3. Section C")
                    print("---")
                    Div = input(">>> Enter the Index of the Division: ")
                    if Div == "1":
                        sec = "A"
                    elif Div == "2":
                            sec = "B"
                    elif Div == "3":
                        sec = "C"
                    else:
                        print("IIError: Invaild Input.")
                    if sec in ["A","B","C"]:
                        query = "Select Rollno, Name from {} where Section =
'{}'".format(tName, sec)
                        mycur.execute(query)
                        Input = mycur.fetchall()
                        mEntry(Input)
                elif eChoice == "3":
                    break
                else:
                    print("IIError: Invalid Input")
        while True:
            print("+"*40)
            print("--- 1. Mark Entry")
            print("--- 2. Academic Report.")
```

```
            print("--- 3. <<< Back")
            print("->")
            aChoice = input("Enter the index of your choice: ")
            if aChoice == "1":
                while True:
                    print("+"*40)
                    print("--- 1. Subjective Assessment.")
                    print("--- 2. Internal Assessment.")
                    print("--- 3. Co-Cirricular Activities")
                    print("--- 4. <<< Back")
                    print("+"*40)
                    ch = input("Enter the index of your choice: ")
                    if ch == "1":
                        Str = "SUB"
                        Redirect()
                    elif ch == "2":
                        Str = "INT"
                        Redirect()
                    elif ch == "3":
                        Str = "CCA"
                        Redirect()
                    elif ch == "4":
                        break
                    else:
                        print("IIError: Invalid Input")
            elif aChoice == "2":
                def Marksheet(Div):
                    print("Cadets' Marksheet")
                    query = 'show tables'
                    mycur.execute(query)
                    data = mycur.fetchall()
                    tablist = []
                    for i in data:
                        if i[0][0:3].upper()== Div:
                            tablist.append(i[0])
                    query = ("select Rollno,Name,Class,Section from cdata")
                    mycur.execute(query)
                    data = mycur.fetchall()
                    header = []
                    for i in tablist:
                        header.append(i[3:].upper())
                    nlist = ["Rollno","Name","Class","Sec"] + header
                    table = PrettyTable(nlist)
                    table.set_style(DOUBLE_BORDER)
                    for k in data:
                        cData = k
                        plist = []
                        for g in tablist:
                            query = ('select * from {} where rollno =
{}').format(g,k[0])
                            mycur.execute(query)
                            aData = mycur.fetchone()
                            tCount = 0
                            Sum = 0
                            if aData != None:
                                for i in range(len(aData)):
                                    if i>3:
                                        if isfloat(aData[i]):
                                            Sum += int(eval(aData[i]))
                                        tCount+=1
                                percentage = Sum/tCount
                                percentage = round(percentage,2)
                                plist.append(percentage)
                            else:
```

```
                                plist.append('AB')
                            tTup = cData + tuple(plist)
                            table.add_row(tTup)
                        print(table)
                    # Menu for Selecting Assessment Type!.
                    print("+"*40)
                    print("--- 1. Subjective Assessment.")
                    print("--- 2. Internal Assessment.")
                    print("--- 3. Co-Cirricular Activities")
                    print("->")
                    ch = input("Enter the Assessment Type to Display Marksheet: ")
                    print("+"*40)
                    if ch == "1":
                        Str = "SUB"
                    elif ch == "2":
                        Str = "INT"
                    elif ch == "3":
                        Str = "CCA"
                    elif ch == "4":
                        break
                    else:
                        print("IIError: Invalid Input")
                    if Str in ["SUB","INT","CCA"]:
                        print("Redirecting", end = "")
                        for i in range(3):
                            t.sleep(1)
                            print(".", end = "")
                        print()
                        print()
                        Marksheet(Str)
                if aChoice == "3":
                    break
        def Report(Key):
            global a,b,c,d,e,block,f,g,h,i,Input,j,k,l,m,Rollno
            Rollno = Key

            Remarks = 'REMARKS:  | > 95 : Brilliant | 85 - 95 : Very Good | 75 - 85 :
Good | \n 65 - 75 : Satisfactory | < 65 : Improvisation Needed! |'
            print()
            print('''The Objective of ICR is to track Student's Progress
Individually, in  order to help  the teachers to be
focused towards the Needed.''')
            print()

            query = "Show Tables"
            mycur.execute(query)
            data = mycur.fetchall()
            tablist = []
            ATD, SUB, INT, CCA = [],[],[],[]
            for i in data:
                tablist.append(i[0])
            for i in tablist:
                if i[0:3].upper() == "ATD":
                    ATD.append(i)
                elif i[0:3].upper() == "SUB":
                    SUB.append(i)
                elif i[0:3].upper() == "INT":
                    INT.append(i)
                elif i[0:3].upper() == "CCA":
                    CCA.append(i)


            atd_List = []
            for k in ATD:
```

```
    string = "Select * from {} where Rollno = '{}'"
    query = string.format(k,Rollno)
    mycur.execute(query)
    aData = mycur.fetchone()
    count = 0
    pcount = 0
    if aData != None:
        for i in range(len(aData)):
            if i > 3:
                if aData[i].upper() == "P":
                    pcount += 1
                count += 1
        percentage = (pcount/count)*100
        percentage = round(percentage, 2)
        atd_List.append(percentage)
    else:
        atd_List.append(0)
if len(atd_List) != 0:
    p_ATD = str(round(sum(atd_List)/len(atd_List), 2))
else:
    p_ATD = "NIL"

int_List = []
for l in INT:
    string = "Select * from {} where Rollno = '{}'"
    query = string.format(l,Rollno)
    mycur.execute(query)
    aData = mycur.fetchone()
    count = 0
    Sum = 0
    if aData != None:
        for i in range(len(aData)):
            if i > 3:
                if isfloat(aData[i]):
                    Sum += int(eval(aData[i]))
                count += 1
        percentage = (Sum/count)
        percentage = round(percentage, 2)
        int_List.append(percentage)
    else:
        int_List.append(0)
if len(int_List) != 0:
    p_INT = str(round(sum(int_List)/len(int_List), 2))
else:
    p_INT = "NIL"


cca_List = []
for m in CCA:
    string = "Select * from {} where Rollno = '{}'"
    query = string.format(m,Rollno)
    mycur.execute(query)
    aData = mycur.fetchone()
    count = 0
    Sum = 0
    if aData != None:
        for i in range(len(aData)):
            if i > 3:
                if isfloat(aData[i]):
                    Sum += int(eval(aData[i]))
                count += 1
        percentage = (Sum/count)
        percentage = round(percentage, 2)
        cca_List.append(percentage)
```

```
    else:
        cca_List.append(0)
if len(cca_List) != 0:
    p_CCA = str(round(sum(cca_List)/len(cca_List), 2))
else:
    p_CCA = "NIL"

query = "select * from cData where Rollno = '{}'".format(Rollno)
mycur.execute(query)
data = mycur.fetchall()


Name, Class, Section = data[0][1], data[0][2], data[0][3]
Length = rectify(len(Name))


SUB.sort()
sub_List = []
header = []
for q in SUB:
    header.append(q[3:].upper())
for g in SUB:
    query = ('select * from {} where rollno = "{}"').format(g,Rollno)
    mycur.execute(query)
    aData = mycur.fetchone()
    tCount = 0
    Sum = 0
    if aData != None:
        for i in range(len(aData)):
            if i>3:
                if isfloat(aData[i]):
                    Sum += int(eval(aData[i]))
                tCount+=1
        percentage = Sum/tCount
        percentage = round(percentage,2)
        sub_List.append(percentage)
    else:
        sub_List.append(0)
if len(sub_List) != 0:
    p_SUB = str(round(sum(sub_List)/len(sub_List), 2))
else:
    p_SUB = "NIL"
input(">>> Press ENTER: ")
print()
print("--- Report Card Generator: Currently Active!")
print()




def verify(cc):
    global n,j,k,l,m,z,logo,nlogo
    try:
        n.destroy()
    except NameError:
        print("",end = "")
    cnfcc = Input.get()
    if cc == cnfcc:
        Input.delete(0, END)
        Input.insert(0,"********")
        for z in [a,b,c,d,e,block,f,g,h,i,Input,j,k,l,m,o]:
            z.destroy()
        Frame = LabelFrame(window, padx = 10, pady = 10, borderwidth = 6)
        frame = LabelFrame(Frame, padx = 10, pady = 10, borderwidth = 3)
```

```
                    Frame.pack()
                    frame.pack()
                    logo = ImageTk.PhotoImage(Image.open(r".\Resources\LogoX.png"))
                    img = Button(frame,image = logo,borderwidth = 0)
                    img.grid(row = 1, column = 1)
                    box = LabelFrame(frame, padx = 10, pady = 10, borderwidth = 0)
                    box.grid(row = 1,column = 2, padx = 10)
                    p = Label(box, text = "SAINIK SCHOOL KALIKIRI", padx = 65)
                    q = Label(box, text = "ANDHRA PRADESH", padx = 65)
                    r = Label(box, text = "sainik.kalikiri@gmail.com", padx = 65)
                    p.pack(),q.pack(),r.pack()
                    nlogo = ImageTk.PhotoImage(Image.open(r".\Resources\nX.png"))
                    Img = Button(frame,image = nlogo,borderwidth = 0)
                    Img.grid(row = 1, column = 3)
                    aFrame = LabelFrame(Frame, padx = 10, pady = 10, borderwidth = 3)
                    aFrame.pack()
                    s = Button(aFrame, text = "Name : {}".format(Name), padx = Length,
bg = "AntiqueWhite2")
                    t = Button(aFrame, text = "Roll.no : {}".format(Rollno), padx = 5,
bg = "gainsboro")
                    u = Button(aFrame, text = "Class : {}".format(Class), padx = 5, bg =
"AntiqueWhite2")
                    v = Button(aFrame, text = "Section : {}".format(Section), padx = 5,
bg = "gainsboro")
                    w = Button(aFrame, text = "Attendance Percentage:
{}%".format(p_ATD), padx = 172, bg = "AntiqueWhite3")
                    s.grid(row = 1, column = 0)
                    t.grid(row = 1, column = 1)
                    u.grid(row = 1, column = 2)
                    v.grid(row = 1, column = 3)
                    w.grid(row = 2, column = 0, columnspan = 4)
                    null0 = Label(Frame, text = "", padx = 95)
                    null0.pack()
                    bFrame = LabelFrame(Frame, padx = 10, pady = 10, borderwidth = 3, bg
= "LemonChiffon3")
                    bFrame.pack()
                    Heading01 = Label(bFrame, text = "Subjective Assessments: ", bg =
"LemonChiffon3")
                    Heading01.grid(row = 0, column = 0, columnspan = len(header))
                    for text1 in header:
                        z = Button(bFrame, text = text1, padx = 16, borderwidth = "3",
bg = "NavajoWhite2")
                        z.grid(row = 1, column = header.index(text1))
                    for text2 in sub_List:
                        z = Button(bFrame, text = str(text2) + "%", padx = 8,
borderwidth = "3", bg = "LemonChiffon2")
                        z.grid(row = 2, column = sub_List.index(text2), columnspan = 1)
                    NULL0 = Label(bFrame, text = " ", bg = "LemonChiffon3")
                    NULL0.grid(row = 3, column = 0, columnspan = len(header))
                    OP = Button(bFrame, text = "Overall Percentage: {}%".format(p_SUB),
borderwidth = 2, bg = "NavajoWhite2")
                    OP.grid(row = 4, column = 0, columnspan = len(header))
                    null1 = Label(Frame, text = "", padx = 95)
                    null1.pack()
                    cFrame = LabelFrame(Frame, padx = 10, pady = 10, borderwidth = 3)
                    cFrame.pack()
                    Internals = Button(cFrame, text = "Internals Score:
{}/100".format(p_INT[:2]), borderwidth = 2, padx = 172, bg = "NavajoWhite3")
                    Internals.pack()
                    cca = Button(cFrame, text = "CCA SCORE:
{}/100".format(int(p_CCA[:2])), borderwidth = 2, padx = 179, bg = "NavajoWhite3")
                    cca.pack()
                    NULL1 = Label(Frame, text = "", padx = 95)
                    NULL1.pack()
```

```
                Text  = Button(Frame, text = Remarks, bg = "misty rose")
                Text.pack()
                desg = Label(window, text = "The Principal, Sainik School
Kalikiri.", anchor = E)
                desg.pack()



            else:
                for z in [j,k,l,m]:
                    z.destroy()
                n = Label(window, text = '''The Captcha Code is Incorrect!
                Try again: ''', 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 [n,k,l,m]:
                    z.pack()

        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 plotter(Key):
    query = "show tables"
    mycur.execute(query)
    data = mycur.fetchall()
    tablist = []
    for i in data:
        if i[0][0:3].upper() == "SUB":
            tablist.append(i[0])
    tablist.sort()
    xBar = []
    for i in tablist:
        xBar.append(i[3:].upper())
    plist = []
    for i in tablist:
        query = "select * from {} where rollno = '{}'".format(i,Key)
        mycur.execute(query)
        aData = mycur.fetchone()
        tCount = 0
        Sum = 0
        if aData != None:
            for i in range(len(aData)):
                if i>3:
                    if isfloat(aData[i]):
                        Sum += int(eval(aData[i]))
                    tCount+=1
            percentage = Sum/tCount
            percentage = round(percentage,2)
            plist.append(percentage)
        else:
            plist.append(0)
    g.plot(xBar, plist, color = "black",
           linestyle = "-.", marker = "o",
           markeredgecolor = "red",
           label = "Examwise Marks plot")
    font = {'family': 'serif',
            'color':  'darkred',
            'weight': 'normal',
            'size': 14,
            }
    font_label = {'family': 'serif',
            'color':  'black',
            'weight': 'normal',
            'size': 10,
            }
    g.title('''ICPlot - Individual Cadet Plot''', fontdict = font)
    g.xlabel("ASSESSMENTS", fontdict = font_label)
    g.ylabel("PERCENTAGES", fontdict = font_label)
    g.legend()
    g.grid()
    g.show()

def remove():
    verify = input("Enter the Administrative Password: ")
    while True:
```

```
                if aPSD == verify:
                     header = ["Rollno", "Name" , "Class", "Section", "Phone number"]
                     table = PrettyTable(header)
                     query = "select Rollno, Name, Class, Section, Phno from cData"
                     mycur.execute(query)
                     data = mycur.fetchall()
                     for i in data:
                          table.add_row(i)
                     table.set_style(DOUBLE_BORDER)
                     print("Reference: ")
                     print(table)
                     print("+"*60)
                     print("---")
                     rno = input("Enter the the Enrollment number of cadet: ")
                     print("--Do you want to continue the process of Deletion: ")
                     ans = input(">>> press [ENTER]: ")
                     if len(ans) == 0:
                          fQuery = "Insert into TCLogs Select * from cdata where Rollno
= '{}'".format(rno)
                          rQuery = "Delete from cdata where Rollno = '{}'".format(rno)
                          mycur.execute(fQuery)
                          mycon.commit()
                          mycur.execute(rQuery)
                          mycon.commit()
                          print("Log Removed...")
                     print("---")
                     print("Press [ENTER] to continue the Removal: ")
                     print("---Else press any key and [Enter]:")
                     prompt = input(">>> ")
                     if len(prompt) != 0:
                          break

     def pAnnouncements():
          print("Announcements Portal.")
          print("---")
          def mAnnounce():
               myfile = open(r".\Resources\Announcements.txt",'a')
               while True:
                    anc = input("Enter a new Announcement: ")
                    myfile.write("~"+anc)
                    myfile.flush()
                    print("Do you want to continue: ")
                    ans01 = input('>>> Press [y] or [n]: ')
                    if ans01 == "y":
                         continue
                    else:
                         break
          def vAnnounce():
               myfile = open(r".\Resources\Announcements.txt",'r')
               while True:
                    print('Press Enter to view:')
                    input('>>>')
                    i = 1
                    record = myfile.read()
                    data =  record.split("~")
                    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 rAnnounce():
            print("-----------------------------------------------")
            print("Instruction: ")
            print("Only one announcement can be removed at a time!.")
            while True:
                myfile = open(r".\Resources\Announcements.txt",'r')
                i = 1
                record = myfile.read()
                data =  record.split("~")
                data.remove(data[0])
                for line in data:
                    print(i, end = ". ")
                    print(line)
                    i += 1
                myfile.close()
                print("---")
                choice = input("Enter the index of Announcement for removal: ")
                if choice.isdigit():
                    cIndex = int(choice)-1
                    data.remove(data[cIndex])
                    myfile = open(r".\Resources\Announcements.txt",'w')
                    header = "Announcements"
                    code = ""
                    for i in data:
                        code += "~" + i
                    flowcode = header + code
                    myfile.write(flowcode)
                    myfile.flush()
                    print("Announcement removed!...")
                else:
                    print("IIError: [Invalid Input]")
                print("To continue the process of removal; press [Enter]")
                ans = input(">>>")
                if len(ans) == 0:
                    continue
                else:
                    myfile.close()
                    break
        while True:
            print("+"*40)
            print("1. Announce")
            print("2. View Announcements")
            print("3. Remove Announcements")
            print("4. <<< Back")
            print("+"*40)
            choice = input("Enter the Index of your Choice: ")
            if choice == '1':
                mAnnounce()
            elif choice == '2':
                vAnnounce()
            elif choice == '3':
                rAnnounce()
            elif choice == '4':
                break
            else:
                print("IIError: Invalid Input")

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

```
                t.sleep(1)
                print(".",end = "")
          print()
          def Update():
                print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~")
                print('Protocol: Username Updation')
                print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~")
                myfile = open(r'.\Resources\tCredentials.dat','rb')
                creds =[]
                try:
                     while True:
                          data = pickle.load(myfile)
                          creds.append(data)
                except EOFError:
                         myfile.close()
                usernames = []
                passwords = []
                for i in creds:
                     usernames.append(i[0])
                     passwords.append(i[1])
                print("+"*40)
                username = input('Enter your Current Username: ')
                nusername = input('Enter your New Username: ')
                if username in usernames:
                     pos = usernames.index(username)
                else:
                     print('IUError: Invalid Username - Please ReCheck')
                     return
                while username.lower()== nusername.lower():
                     print('Username matches with the previous one!')
                     print('- Try changing it into different one')
                     print('If you want to retain your previous username: ')
                     print('- Cancel the process by Entering "Quit" below.' )
                     print("->")
                     nusername = input('Enter your New Username: ')
                if nusername.upper()=='QUIT':
                     return
                print("---")
                print('Verification protocol: ')
                print('>>> Confirmation to change the Username: ')
                vPSD = input('Enter Password: ')
                print("---")
                cc = captchaGen()
                print('Captcha code: ',cc)
                cnfcc = input('Enter the 8 character Captcha Code shown above: ')
                if cnfcc == cc:
                     if vPSD == passwords[pos]:
                          print("--- Authentication Successful ---")
                          print("Attempting To Change Username", end = "")
                          for i in range(5):
                               t.sleep(1)
                               print(".",end = "")
                          print()
                          creds[pos][0]= nusername
                          myfile=open(r'.\Resources\tCredentials.dat','wb')
                          for i in creds:
                               pickle.dump(i,myfile)
                          myfile.flush()
                          print("Username Updated Successfully...")
                          myfile.close()
                     else:
                          print('Access Denied: Unauthorised attempt for changing
Username! | Check for the correct Credentials.')
                else:
```

```
                print('CCError: Invalid Input for Captcha Code')
        def Change():
            print("~~~~~~~~~~~~~~~~~~~~~~~~~~~")
            print('Protocol: Password Updation')
            print("~~~~~~~~~~~~~~~~~~~~~~~~~~~")
            myfile = open(r'.\Resources\tCredentials.dat','rb')
            creds=[]
            try:
                while True:
                    data=pickle.load(myfile)
                    creds.append(data)
            except EOFError:
                    myfile.close()
            usernames=[]
            passwords=[]
            for i in creds:
                usernames.append(i[0])
                passwords.append(i[1])
            print("+"*40)
            username = input('Enter your Username: ')
            password = input("Enter Password: ")
            if username in usernames:
                pos = usernames.index(username)
                if passwords[pos] == password:
                    print("--- Authentication Successful ---")
                    print("+"*40)
                    psd = input("Enter your New Password: ")
                    while password.lower()== psd.lower():
                        print('Password matches with the previous one!')
                        print('- Try changing it into different one')
                        print('If you want to retain your previous Password: ')
                        print('- Cancel the process by Entering "Quit" below.' )
                        print("->")
                        psd = input('Enter your New Password: ')
                    if psd.upper()=='QUIT':
                        return
                    cnfpsd = input("Retype the New Password: ")
                    if psd == cnfpsd:
                        print("---")
                        cc = captchaGen()
                        print('Captcha code: ',cc)
                        cnfcc = input('Enter the 8 character Captcha Code shown
above: ')
                        if cnfcc == cc:
                            print("Attempting To Change Password", end = "")
                            for i in range(5):
                                t.sleep(1)
                                print(".",end = "")
                            print()
                            creds[pos][1] = psd
                            myfile=open(r'.\Resources\tCredentials.dat','wb')
                            for i in creds:
                                pickle.dump(i,myfile)
                            myfile.flush()
                            print(">>> Password Changed Successfully...")
                            myfile.close()
                        else:
                            print('CCError: Invalid Input for Captcha Code')
                            return
                    else:
                        print('IPError: Passwords did not match.')
                        return
                else:
                    print('Access Denied: Unauthorised attempt for changing
```

```
Password! | Check for the correct Credentials.')
                    return
               else:
                    print('IUError: Invalid Username - Please ReCheck')
                    return

          def Delete():
               print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~")
               print("Protocol: Account Deletion.")
               print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~")
               myfile =open(r'.\Resources\tCredentials.dat','rb')
               creds=[]
               try:
                    while True:
                         data=pickle.load(myfile)
                         creds.append(data)
               except EOFError:
                        myfile.close()
               usernames=[]
               passwords=[]
               for i in creds:
                    usernames.append(i[0])
                    passwords.append(i[1])
               user =input('Enter your Username for Deletion: ')
               psd =input('Enter your Password: ')
               if user in usernames:
                    pos = usernames.index(user)
                    if passwords[pos] == psd:
                         print("--- Authentication Successful ---")
                         print("+"*40)
                         cc = captchaGen()
                         print('Captcha code: ',cc)
                         cnfcc = input('Enter the 8 character Captcha Code shown
above: ')
                         if cnfcc == cc:
                              print("Attempting To Delete Account", end = "")
                              for i in range(5):
                                   t.sleep(1)
                                   print(".",end = "")
                              print()
                              log = creds[pos]
                              creds.remove(log)
                              myfile=open(r'.\Resources\tCredentials.dat','wb')
                              for i in creds:
                                   pickle.dump(i,myfile)
                              myfile.flush()
                              print(">>> Account Deleted Successfully...")
                              myfile.close()
                         else:
                              print('CCError: Invalid Input for Captcha Code')
                    else:
                         print('Access Denied: Unauthorised attempt for Deleting
Account! | Check for the correct Credentials.')
               else:
                    print('IUError: Invalid Username - Please ReCheck')


          while True:
               print("+"*40)
               print('--- 1.Update username')
               print('--- 2.Change password')
               print('--- 3.Delete Account')
               print('--- 4. Back')
               print("->")
```

```
                psChoice = input("Enter the Index of your Requirement: ")
                if psChoice == "1":
                     Update()
                elif psChoice == "2":
                     Change()
                elif psChoice == "3":
                     Delete()
                elif psChoice == "4":
                     break
                else:
                     print("IIError: Invalid Input")

     def Admin():
          verify = input("Enter the Administrative Password to continue for Privileged
Operations: ")
          if verify == aPSD:
               while True:
                  print("="*20)
                  print("  Menu: ")
                  print("- 1. User Lookup")
                  print("- 2. Change Password")
                  print("- 3. <- Back")
                  print("->")
                  oChoice = input(">>> Enter your Choice: ")
                  if oChoice == "1":
                      print("="*20)
                      print("  Menu: ")
                      print("- 1. Manual Search")
                      print("- 2. Auto Search")
                      print("- 3. <- Back")
                      print("->")
                      sChoice = input(">>> Enter your Choice: ")
                      if sChoice == "1":
                          print("+"*40)
                          myfile = open(r".\Resources\sCredentials.dat", "rb")
                          creds = []
                          try:
                              while True:
                                   data = pickle.load(myfile)
                                   creds.append(data)
                          except EOFError:
                                   myfile.close()
                          print("--- Loading Credentials", end = "")
                          for i in range(3):
                              t.sleep(1)
                              print(".", end = "")
                          print()
                          print("Credentials: ")
                          table =
PrettyTable(["Username","Password","Rollno","UserType","Account","Security"])
                          table.set_style(DOUBLE_BORDER)
                          for i in creds:
                              if len(i) == 2:
                                  i.append("000")
                              if len(i) == 3:
                                  table.add_row(i+["Standard","Student","Binary"])
                          print(table)
                      elif sChoice == "2":
                          myfile = open(r".\Resources\sCredentials.dat", "rb")
                          creds = []
                          try:
                              while True:
                                   data = pickle.load(myfile)
                                   creds.append(data)
```

```
                        except EOFError:
                                 myfile.close()
                        print("--- Loading Credentials", end = "")
                        print()
                        for i in range(3):
                            t.sleep(1)
                            print(".", end = "")
                        print()
                        usr = input("Enter your Username to search for Credentials:
")
                        count = 0
                        for i in creds:
                            if i[0] == usr:
                                count = 1
                                Record = i
                        if count == 0:
                            print("Username Not Found!!")
                        else:
                            table =
PrettyTable(["Username","Password","Rollno","UserType","Account","Security"])
                            table.set_style(DOUBLE_BORDER)
                            if len(Record) == 2:
                                Record.append("000")
                            if len(Record) == 3:
                                table.add_row(Record+
["Standard","Student","Binary"])
                                print("Credentials: ")
                                print(table)

                elif oChoice == "2":
                    myfile = open(r'.\Resources\sCredentials.dat','rb')
                    creds=[]
                    try:
                        while True:
                             data=pickle.load(myfile)
                             creds.append(data)
                    except EOFError:
                            myfile.close()
                    usernames=[]
                    passwords=[]
                    for i in creds:
                        usernames.append(i[0])
                        passwords.append(i[1])
                    print("+"*40)
                    username = input('Enter your Username: ')
                    if username in usernames:
                        pos = usernames.index(username)
                        print("--- Intrusion: Authentication Overwrite Successful --
-")
                        print("+"*40)
                        psd = input("Enter your New Password: ")
                        cnfpsd = input("Retype the New Password: ")
                        if psd == cnfpsd:
                            print("---")
                            cc = captchaGen()
                            print('Captcha code: ',cc)
                            cnfcc = input('Enter the 8 character Captcha Code shown
above: ')
                            if cnfcc == cc:
                                print("Attempting To Change Password", end = "")
                                for i in range(5):
                                    t.sleep(1)
                                    print(".",end = "")
                                print()
```

```
                                creds[pos][1] = psd
                                myfile=open(r'.\Resources\sCredentials.dat','wb')
                                for i in creds:
                                    pickle.dump(i,myfile)
                                myfile.flush()
                                print(">>> Password Changed Successfully...")
                                myfile.close()
                            else:
                                print('CCError: Invalid Input for Captcha Code')
                                return
                        else:
                            print('IPError: Passwords did not match.')
                            return
                    else:
                        print('IUError: Invalid Username - Please ReCheck')
                elif oChoice == "3":
                    break
        else:
             print("============== ! Unauthorised Intrusion Attempted !
==============")

    while True:
        print("+"*40)
        print("--- 1.  Enroll New Cadet")
        print("--- 2.  Update Logs")
        print("--- 3.  Attendance")
        print("--- 4.  Academics")
        print("--- 5.  ICReport")
        print("--- 6.  Remove Logs")
        print("--- 7.  Announcements")
        print("--- 8.  Account Settings")
        print("--- 9.  Admin Operations")
        print("--- 10. LogOut")
        print("+"*40)
        choice = input("Enter the Index of your Choice: ")
        if choice == '1':
             enroll()
        elif choice == '2':
             update()
        elif choice == '3':
             pAttendance()
        elif choice == '4':
             pAcademics()
        elif choice == '5':
             global Rollno,prompt_001,prompt_002,prompt_003
             Rollno = input("Enter the Cadet's Enrollment Number: ")
             prompt_001 = None
             prompt_002 = None
             prompt_003 = None
             def pPerformance(accessKey):
                 def try_TITLE():
                     global prompt_001,prompt_002,prompt_003
                     if prompt_001 == None:
                         print("Subjective Assessments: ")
                         prompt_001 = "Complete"
                     elif prompt_002 == None:
                         print("Internal Assessments: ")
                         prompt_002 = "Complete"
                     elif prompt_003 == None:
                         print("CCA Assessments: ")
                         prompt_003 = "Complete"

                 def redirect(i):
                     print("--- Exam Name: ", end = "")
```

```
                print(i[3:].upper())
                print("->")
                header = []
                query = "desc {}".format(i)
                mycur.execute(query)
                data = mycur.fetchall()
                for j in data:
                    header.append(j[0].title())
                table = PrettyTable(header)
                table.set_style(DOUBLE_BORDER)
                string = "Select * from {} where Rollno = {}"
                query = string.format(i, accessKey)
                mycur.execute(query)
                Record = mycur.fetchall()
                for i in Record:
                    table.add_row(i)
                print(table)

            query = "Show tables"
            mycur.execute(query)
            data = mycur.fetchall()
            tablist = []
            for i in data:
                if i[0][0:3].upper() in ["SUB","INT","CCA"]:
                    tablist.append(i[0])
            tablist.sort(reverse = True)
            print("--- Loading all Marksheets", end = "")
            for i in range(3):
                t.sleep(1)
                print(".", end = "")
            print()
            for i in tablist:
                if i[0:3].upper() == "SUB":
                    if prompt_001 != "Complete":
                        try_TITLE()
                    redirect(i)
                elif i[0:3].upper() == "INT":
                    if prompt_002 != "Complete":
                        try_TITLE()
                    redirect(i)
                elif i[0:3].upper() == "CCA":
                    if prompt_002 != "Complete":
                        try_TITLE()
                    redirect(i)
        pPerformance(Rollno)
        while True:
             print("==============================")
             print(" Menu: ")
             print("--- 1. Show Development Plots")
             print("--- 2. Generate Report Card")
             print("--- 3. <- Back")
             Ans = input(">>> Enter your Choice: ")
             if Ans == "1":
                  plotter(Rollno)
             elif Ans == "2":
                  Report(Rollno)
             elif Ans == "3":
                  break
             else:
                  print("Invalid Input")

    elif choice == '6':
         remove()
    elif choice == '7':
```

```
                pAnnouncements()
            elif choice == '8':
                profileSet()
            elif choice == "9":
                Admin()
            elif choice == '10':
                print()
                print("-="*15, "LOGGED OUT","=-"*15)
                print()
                break
            else:
                print("IIError: Invalid Input")
########################################################################################
############
def sPortal(User, Rollno):
    print("_"*72)
    print("~"*72)
    print("=-"*14, 'Student Portal', "-="*14)
    print(">>> Welcome",User)
    print("---")
    accessKey = Rollno
    def ACD_Performance():
        global prompt_001,prompt_002,prompt_003
        prompt_001 = None
        prompt_002 = None
        prompt_003 = None
        def pPerformance(accessKey):
            def try_TITLE():
                global prompt_001,prompt_002,prompt_003
                if prompt_001 == None:
                    print("Subjective Assessments: ")
                    prompt_001 = "Complete"
                elif prompt_002 == None:
                    print("Internal Assessments: ")
                    prompt_002 = "Complete"
                elif prompt_003 == None:
                    print("CCA Assessments: ")
                    prompt_003 = "Complete"

            def redirect(i):
                print("--- Exam Name: ", end = "")
                print(i[3:].upper())
                print("->")
                header = []
                query = "desc {}".format(i)
                mycur.execute(query)
                data = mycur.fetchall()
                for j in data:
                    header.append(j[0].title())
                table = PrettyTable(header)
                table.set_style(DOUBLE_BORDER)
                string = "Select * from {} where Rollno = {}"
                query = string.format(i, accessKey)
                mycur.execute(query)
                Record = mycur.fetchall()
                for i in Record:
                    table.add_row(i)
                print(table)

            query = "Show tables"
            mycur.execute(query)
            data = mycur.fetchall()
            tablist = []
            for i in data:
```

```
                if i[0][0:3].upper() in ["SUB","INT","CCA"]:
                    tablist.append(i[0])
            tablist.sort(reverse = True)
            print("--- Loading all Marksheets", end = "")
            for i in range(3):
                t.sleep(1)
                print(".", end = "")
            print()
            for i in tablist:
                if i[0:3].upper() == "SUB":
                    if prompt_001 != "Complete":
                        try_TITLE()
                    redirect(i)
                elif i[0:3].upper() == "INT":
                    if prompt_002 != "Complete":
                        try_TITLE()
                    redirect(i)
                elif i[0:3].upper() == "CCA":
                    if prompt_002 != "Complete":
                        try_TITLE()
                    redirect(i)

        pPerformance(accessKey)

    def Report():
        global a,b,c,d,e,block,f,g,h,i,Input,j,k,l,m

        Remarks = 'REMARKS:  | > 95 : Brilliant | 85 - 95 : Very Good | 75 - 85 :
Good | \n 65 - 75 : Satisfactory | < 65 : Improvisation Needed! |'
        print()
        print('''The Objective of ICR is to track Student's Progress
Individually, in  order to help  the teachers to be
focused towards the Needed.''')
        print()

        query = "Show Tables"
        mycur.execute(query)
        data = mycur.fetchall()
        tablist = []
        ATD, SUB, INT, CCA = [],[],[],[]
        for i in data:
            tablist.append(i[0])
        for i in tablist:
            if i[0:3].upper() == "ATD":
                ATD.append(i)
            elif i[0:3].upper() == "SUB":
                SUB.append(i)
            elif i[0:3].upper() == "INT":
                INT.append(i)
            elif i[0:3].upper() == "CCA":
                CCA.append(i)


        atd_List = []
        for k in ATD:
            string = "Select * from {} where Rollno = '{}'"
            query = string.format(k,Rollno)
            mycur.execute(query)
            aData = mycur.fetchone()
            count = 0
            pcount = 0
            if aData != None:
                for i in range(len(aData)):
                    if i > 3:
```

```
                if aData[i].upper() == "P":
                    pcount += 1
                count += 1
        percentage = (pcount/count)*100
        percentage = round(percentage, 2)
        atd_List.append(percentage)
    else:
        atd_List.append(0)
if len(atd_List) != 0:
    p_ATD = str(round(sum(atd_List)/len(atd_List), 2))
else:
    p_ATD = "NIL"

int_List = []
for l in INT:
    string = "Select * from {} where Rollno = '{}'"
    query = string.format(l,Rollno)
    mycur.execute(query)
    aData = mycur.fetchone()
    count = 0
    Sum = 0
    if aData != None:
        for i in range(len(aData)):
            if i > 3:
                if isfloat(aData[i]):
                    Sum += int(eval(aData[i]))
                count += 1
        percentage = (Sum/count)
        percentage = round(percentage, 2)
        int_List.append(percentage)
    else:
        int_List.append(0)
if len(int_List) != 0:
    p_INT = str(round(sum(int_List)/len(int_List), 2))
else:
    p_INT = "NIL"


cca_List = []
for m in CCA:
    string = "Select * from {} where Rollno = '{}'"
    query = string.format(m,Rollno)
    mycur.execute(query)
    aData = mycur.fetchone()
    count = 0
    Sum = 0
    if aData != None:
        for i in range(len(aData)):
            if i > 3:
                if isfloat(aData[i]):
                    Sum += int(eval(aData[i]))
                count += 1
        percentage = (Sum/count)
        percentage = round(percentage, 2)
        cca_List.append(percentage)
    else:
        cca_List.append(0)
if len(cca_List) != 0:
    p_CCA = str(round(sum(cca_List)/len(cca_List), 2))
else:
    p_CCA = "NIL"

query = "select * from cData where Rollno = '{}'".format(Rollno)
mycur.execute(query)
```

```
data = mycur.fetchall()


Name, Class, Section = data[0][1], data[0][2], data[0][3]
Length = rectify(len(Name))


SUB.sort()
sub_List = []
header = []
for q in SUB:
    header.append(q[3:].upper())
for g in SUB:
    query = ('select * from {} where rollno = "{}"').format(g,Rollno)
    mycur.execute(query)
    aData = mycur.fetchone()
    tCount = 0
    Sum = 0
    if aData != None:
        for i in range(len(aData)):
            if i>3:
                if isfloat(aData[i]):
                    Sum += int(eval(aData[i]))
                tCount+=1
        percentage = Sum/tCount
        percentage = round(percentage,2)
        sub_List.append(percentage)
    else:
        sub_List.append(0)
if len(sub_List) != 0:
    p_SUB = str(round(sum(sub_List)/len(sub_List), 2))
else:
    p_SUB = "NIL"
input(">>> Press ENTER: ")
print()
print("--- Report Card Generator: Currently Active!")
print()




def verify(cc):
    global n,j,k,l,m,z,logo,nlogo
    try:
        n.destroy()
    except NameError:
        print("",end = "")
    cnfcc = Input.get()
    if cc == cnfcc:
        Input.delete(0, END)
        Input.insert(0,"********")
        for z in [a,b,c,d,e,block,f,g,h,i,Input,j,k,l,m,o]:
            z.destroy()
        Frame = LabelFrame(window, padx = 10, pady = 10, borderwidth = 6)
        frame = LabelFrame(Frame, padx = 10, pady = 10, borderwidth = 3)
        Frame.pack()
        frame.pack()
        logo = ImageTk.PhotoImage(Image.open(r".\Resources\LogoX.png"))
        img = Button(frame,image = logo,borderwidth = 0)
        img.grid(row = 1, column = 1)
        box = LabelFrame(frame, padx = 10, pady = 10, borderwidth = 0)
        box.grid(row = 1,column = 2, padx = 10)
        p = Label(box, text = "SAINIK SCHOOL KALIKIRI", padx = 65)
        q = Label(box, text = "ANDHRA PRADESH", padx = 65)
```

```
                    r = Label(box, text = "sainik.kalikiri@gmail.com", padx = 65)
                    p.pack(),q.pack(),r.pack()
                    nlogo = ImageTk.PhotoImage(Image.open(r".\Resources\nX.png"))
                    Img = Button(frame,image = nlogo,borderwidth = 0)
                    Img.grid(row = 1, column = 3)
                    aFrame = LabelFrame(Frame, padx = 10, pady = 10, borderwidth = 3)
                    aFrame.pack()
                    s = Button(aFrame, text = "Name : {}".format(Name), padx = Length,
bg = "AntiqueWhite2")
                    t = Button(aFrame, text = "Roll.no : {}".format(Rollno), padx = 5,
bg = "gainsboro")
                    u = Button(aFrame, text = "Class : {}".format(Class), padx = 5, bg =
"AntiqueWhite2")
                    v = Button(aFrame, text = "Section : {}".format(Section), padx = 5,
bg = "gainsboro")
                    w = Button(aFrame, text = "Attendance Percentage:
{}%".format(p_ATD), padx = 172, bg = "AntiqueWhite3")
                    s.grid(row = 1, column = 0)
                    t.grid(row = 1, column = 1)
                    u.grid(row = 1, column = 2)
                    v.grid(row = 1, column = 3)
                    w.grid(row = 2, column = 0, columnspan = 4)
                    null0 = Label(Frame, text = "", padx = 95)
                    null0.pack()
                    bFrame = LabelFrame(Frame, padx = 10, pady = 10, borderwidth = 3, bg
= "LemonChiffon3")
                    bFrame.pack()
                    Heading01 = Label(bFrame, text = "Subjective Assessments: ", bg =
"LemonChiffon3")
                    Heading01.grid(row = 0, column = 0, columnspan = len(header))
                    for text1 in header:
                        z = Button(bFrame, text = text1, padx = 16, borderwidth = "3",
bg = "NavajoWhite2")
                        z.grid(row = 1, column = header.index(text1))
                    for text2 in sub_List:
                        z = Button(bFrame, text = str(text2) + "%", padx = 8,
borderwidth = "3", bg = "LemonChiffon2")
                        z.grid(row = 2, column = sub_List.index(text2), columnspan = 1)
                    NULL0 = Label(bFrame, text = " ", bg = "LemonChiffon3")
                    NULL0.grid(row = 3, column = 0, columnspan = len(header))
                    OP = Button(bFrame, text = "Overall Percentage: {}%".format(p_SUB),
borderwidth = 2, bg = "NavajoWhite2")
                    OP.grid(row = 4, column = 0, columnspan = len(header))
                    null1 = Label(Frame, text = "", padx = 95)
                    null1.pack()
                    cFrame = LabelFrame(Frame, padx = 10, pady = 10, borderwidth = 3)
                    cFrame.pack()
                    Internals = Button(cFrame, text = "Internals Score:
{}/100".format(p_INT[:2]), borderwidth = 2, padx = 172, bg = "NavajoWhite3")
                    Internals.pack()
                    cca = Button(cFrame, text = "CCA SCORE:
{}/100".format(int(p_CCA[:2])), borderwidth = 2, padx = 179, bg = "NavajoWhite3")
                    cca.pack()
                    NULL1 = Label(Frame, text = "", padx = 95)
                    NULL1.pack()
                    Text  = Button(Frame, text = Remarks, bg = "misty rose")
                    Text.pack()
                    desg = Label(window, text = "The Principal, Sainik School
Kalikiri.", anchor = E)
                    desg.pack()
```

```
            else:
                for z in [j,k,l,m]:
                    z.destroy()
                n = Label(window, text = '''The Captcha Code is Incorrect!
                Try again: ''', 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 [n,k,l,m]:
                    z.pack()

        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("~")
          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):
         t.sleep(1)
         print(".",end = "")
    print()
    def Update():
         print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~")
         print('Protocol: Username Updation')
         print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~")
         myfile = open(r'.\Resources\sCredentials.dat','rb')
         creds =[]
         try:
              while True:
                   data = pickle.load(myfile)
                   creds.append(data)
         except EOFError:
                  myfile.close()
         usernames = []
         passwords = []
         for i in creds:
              usernames.append(i[0])
              passwords.append(i[1])
         print("+"*40)
         username = input('Enter your Current Username: ')
         nusername = input('Enter your New Username: ')
         if username in usernames:
              pos = usernames.index(username)
         else:
              print('IUError: Invalid Username - Please ReCheck')
              return
         while username.lower()== nusername.lower():
              print('Username matches with the previous one!')
              print('- Try changing it into different one')
              print('If you want to retain your previous username: ')
              print('- Cancel the process by Entering "Quit" below.' )
              print("->")
              nusername = input('Enter your New Username: ')
         if nusername.upper()=='QUIT':
              return
         print("---")
```

```
            print('Verification protocol: ')
            print('>>> Confirmation to change the Username: ')
            vPSD = input(r'Enter Password: ')
            print("---")
            cc = captchaGen()
            print('Captcha code: ',cc)
            cnfcc = input('Enter the 8 character Captcha Code shown above: ')
            if cnfcc == cc:
                 if vPSD == passwords[pos]:
                      print("--- Authentication Successful ---")
                      print("Attempting To Change Username", end = "")
                      for i in range(5):
                           t.sleep(1)
                           print(".",end = "")
                      print()
                      creds[pos][0]= nusername
                      myfile=open(r'.\Resources\sCredentials.dat','wb')
                      for i in creds:
                           pickle.dump(i,myfile)
                      myfile.flush()
                      print("Username Updated Successfully...")
                      myfile.close()
                 else:
                      print('Access Denied: Unauthorised attempt for changing
Username! | Check for the correct Credentials.')
            else:
                 print('CCError: Invalid Input for Captcha Code')
       def Change():
            print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~")
            print('Protocol: Password Updation')
            print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~")
            myfile = open(r'.\Resources\sCredentials.dat','rb')
            creds=[]
            try:
                 while True:
                      data=pickle.load(myfile)
                      creds.append(data)
            except EOFError:
                     myfile.close()
            usernames=[]
            passwords=[]
            for i in creds:
                 usernames.append(i[0])
                 passwords.append(i[1])
            print("+"*40)
            username = input('Enter your Username: ')
            password = input("Enter Password: ")
            if username in usernames:
                 pos = usernames.index(username)
                 if passwords[pos] == password:
                      print("--- Authentication Successful ---")
                      print("+"*40)
                      psd = input("Enter your New Password: ")
                      while password.lower()== psd.lower():
                           print('Password matches with the previous one!')
                           print('- Try changing it into different one')
                           print('If you want to retain your previous Password: ')
                           print('- Cancel the process by Entering "Quit" below.' )
                           print("->")
                           psd = input('Enter your New Password: ')
                      if psd.upper()=='QUIT':
                           return
                      cnfpsd = input("Retype the New Password: ")
                      if psd == cnfpsd:
```

```
                              print("---")
                              cc = captchaGen()
                              print('Captcha code: ',cc)
                              cnfcc = input('Enter the 8 character Captcha Code shown
above: ')
                              if cnfcc == cc:
                                   print("Attempting To Change Password", end = "")
                                   for i in range(5):
                                        t.sleep(1)
                                        print(".",end = "")
                                   print()
                                   creds[pos][1] = psd
                                   myfile=open(r'.\Resources\sCredentials.dat','wb')
                                   for i in creds:
                                        pickle.dump(i,myfile)
                                   myfile.flush()
                                   print(">>> Password Changed Successfully...")
                                   myfile.close()
                              else:
                                   print('CCError: Invalid Input for Captcha Code')
                                   return
                         else:
                              print('IPError: Passwords did not match.')
                              return
                    else:
                         print('Access Denied: Unauthorised attempt for changing
Password! | Check for the correct Credentials.')
                         return
               else:
                    print('IUError: Invalid Username - Please ReCheck')
                    return

          def Delete():
               print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~")
               print("Protocol: Account Deletion.")
               print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~")
               myfile =open(r'.\Resources\sCredentials.dat','rb')
               creds=[]
               try:
                    while True:
                         data=pickle.load(myfile)
                         creds.append(data)
               except EOFError:
                        myfile.close()
               usernames=[]
               passwords=[]
               for i in creds:
                    usernames.append(i[0])
                    passwords.append(i[1])
               user =input('Enter your Username for Deletion: ')
               psd =input('Enter your Password: ')
               if user in usernames:
                    pos = usernames.index(user)
                    if passwords[pos] == psd:
                         print("--- Authentication Successful ---")
                         print("+"*40)
                         cc = captchaGen()
                         print('Captcha code: ',cc)
                         cnfcc = input('Enter the 8 character Captcha Code shown above:
')
                         if cnfcc == cc:
                              print("Attempting To Delete Account", end = "")
                              for i in range(5):
                                   t.sleep(1)
```

```
                                   print(".",end = "")
                              print()
                              log = creds[pos]
                              creds.remove(log)
                              myfile=open(r'.\Resources\sCredentials.dat','wb')
                              for i in creds:
                                   pickle.dump(i,myfile)
                              myfile.flush()
                              print(">>> Account Deleted Successfully...")
                              myfile.close()
                         else:
                              print('CCError: Invalid Input for Captcha Code')
                    else:
                         print('Access Denied: Unauthorised attempt for Deleting
Account! | Check for the correct Credentials.')
               else:
                    print('IUError: Invalid Username - Please ReCheck')

          while True:
               print("+"*40)
               print('--- 1.Update username')
               print('--- 2.Change password')
               print('--- 3.Delete Account')
               print('--- 4. Back')
               print("->")
               psChoice = input("Enter the Index of your Requirement: ")
               if psChoice == "1":
                    Update()
               elif psChoice == "2":
                    Change()
               elif psChoice == "3":
                    Delete()
               elif psChoice == "4":
                    break
               else:
                    print("IIError: Invalid Input")
     while True:
          print("+"*40)
          print("--- 1. Academic Performance")
          print("--- 2. Report Card")
          print("--- 3. Announcements")
          print("--- 4. Account Settings")
          print("--- 5. LogOut")
          print("+"*40)
          choice = input("Enter the Index of your Choice: ")
          if choice == '1':
               ACD_Performance()
          elif choice == '2':
               Report()
          elif choice == '3':
               pAnnouncements()
          elif choice == '4':
               profileSet()
          elif choice == '5':
               print()
               print("-="*15, "LOGGED OUT","=-"*15)
               print()
               break
          else:
               print("IIError: Invalid Input")
#################################################################################
############
print("\t\t _____________________________________ ")
print("\t\t|                                     |")
```

```
print("\t\t| * * * * *    SCHOOL    * * * * * |")
print("\t\t|  * * * *   MANAGEMENT   * * * *  |")
print("\t\t| * * * * *    SYSTEM    * * * * * |")
print("\t\t|_____________________________________|")
print()
print("*"*75)
print("---")
input("Press Enter to Continue: ")
while True:
     print("+"*40)
     print("--- 1. LogIn")
     print("--- 2. Exit")
     print("->")
     Log = input('Enter your Choice: ')
     if Log == "1":
          while True:
               print("+"*40)
               print("1. Teacher Login")
               print("2. Student Login")
               print("3. <- Back")
               print("---")
               Login_mode01 = input("Enter your choice (1 or 2): ")
               if Login_mode01 == '1':
                    while True:
                         print("+"*40)
                         print('--- 1. Login')
                         print('--- 2. Register')
                         print("--- 3. <- Back")
                         print("-> ")
                         Login_mode02 = input("Enter your choice (1 or 2): ")
                         myfile = open(r".\Resources\tCredentials.dat",'rb')
                         creds = []
                         try:
                              while True:
                                   data = pickle.load(myfile)
                                   creds.append(data)
                         except EOFError:
                                   myfile.close()
                         usernames = []
                         passwords = []
                         for i in creds:
                              usernames.append(i[0])
                              passwords.append(i[1])
                         if Login_mode02 == '1':
                              print("+"*40)
                              usr = input("Enter Username: ")
                              psd = input("Enter Password: ")
                              if usr in usernames:
                                   pos = usernames.index(usr)
                                   if passwords[pos] == psd:
                                        print(">>> Authentication Successful <<<")
                                        print("--- Loading Portal", end = "")
                                        for i in range(5):
                                            print(".",end = "")
                                            t.sleep(1)
                                        print()
                                        tPortal(usr)
                                   else:
                                        print('IPError: Invalid Password entered.')
                              else:
                                   print('IUError: Invalid Username - Please ReCheck')
                         elif Login_mode02 == '2':
                              myfile = open(r".\Resources\tCredentials.dat",'ab')
                              print("+"*40)
```

```
                              verify = input("Enter the Administrative password: ")
                              print("---")
                              if aPSD == verify:
                                   usr = input('Enter a Username: ')
                                   if usr not in usernames:
                                        psd = input('Enter the Password: ')
                                        cnfpsd = input('Confirm your Password: ')
                                        print("+"*40)
                                        cc = captchaGen()
                                        print("Captcha Code:" , end = "    ")
                                        print(cc)
                                        cnfcc = input('Enter the 8 character Captcha
Code shown above: ')
                                        print("+"*40)
                                        if cc == cnfcc:
                                             if psd == cnfpsd:
                                                  List = [usr,psd]
                                                  pickle.dump(List, myfile)
                                                  myfile.flush()
                                                  print("--- Adding User", end = "")
                                                  for i in range(5):
                                                      print(".",end = "")
                                                      t.sleep(1)
                                                  print()
                                                  print('User added successfully.')
                                                  print('- Please Re-login to
continue...')
                                             else:
                                                  print('IPError: Passwords did not
match.')
                                        else:
                                             print('CCError: Invalid Input for Captcha
Code')
                                   else:
                                        print('User with this username already exists:
Please try again...')
                              else:
                                   print('Access Denied: Unauthorised attempt for
Login! | Check for the correct Credentials.')
                         elif Login_mode02 == "3":
                              break
                         else:
                              print("Invalid Input")
               elif Login_mode01 == '2':
                    while True:
                         print("+"*40)
                         print('--- 1. Login')
                         print('--- 2. Register')
                         print("--- 3. <- Back")
                         print("->")
                         Login_mode02 = input("Enter your choice (1 or 2): ")
                         myfile = open(r".\Resources\sCredentials.dat",'rb')
                         creds = []
                         try:
                              while True:
                                   data = pickle.load(myfile)
                                   creds.append(data)
                         except EOFError:
                                   myfile.close()
                         usernames = []
                         passwords = []
                         for i in creds:
                              usernames.append(i[0])
                              passwords.append(i[1])
```

```
                    if Login_mode02 == '1':
                         print("+"*40)
                         usr = input("Enter Username: ")
                         psd = input("Enter Password: ")
                         if usr in usernames:
                              pos = usernames.index(usr)
                              if passwords[pos] == psd:
                                   try:
                                        Rno = creds[pos][2]
                                   except IndexError:
                                        Rno = ""
                                   print(">>> Authentication Successful <<<")
                                   print("--- Loading Portal", end = "")
                                   for i in range(5):
                                       print(".",end = "")
                                       t.sleep(1)
                                   print()
                                   sPortal(usr,Rno)
                              else:
                                   print('IPError: Invalid Password entered.')
                         else:
                              print('IUError: Invalid Username - Please ReCheck')
                    elif Login_mode02 == '2':
                         myfile = open(r".\Resources\sCredentials.dat",'ab')
                         print("+"*40)
                         verify = input("Enter your Roll number: ")
                         query = ("select Rollno from cDATA")
                         mycur.execute(query)
                         data = mycur.fetchall()
                         record = []
                         for i in data:
                              record.append(i[0])
                         print("---")
                         if verify in record:
                              print("+"*40)
                              usr = input('Enter a Username: ')
                              if usr not in usernames:
                                   psd = input('Enter the Password: ')
                                   cnfpsd = input('Confirm your Password: ')
                                   print("+"*40)
                                   cc = captchaGen()
                                   print("Captcha Code:" , end = "    ")
                                   print(cc)
                                   cnfcc = input('Enter the 8 character Captcha
Code shown above: ')
                                   print("+"*40)
                                   if cc == cnfcc:
                                        if psd == cnfpsd:
                                             List = [usr,psd,verify]
                                             pickle.dump(List, myfile)
                                             myfile.flush()
                                             print("--- Adding User", end = "")
                                             for i in range(5):
                                                 print(".",end = "")
                                                 t.sleep(1)
                                             print()
                                             print('User added successfully.')
                                             print('- Please Re-login to
continue...')
                                        else:
                                             print('IPError: Passwords did not
match.')
                                   else:
                                        print('CCError: Invalid Input for Captcha
```

```
Code')
                                        else:
                                             print('User with this username already exists:
Please try again...')
                                   else:
                                        print('The cadet with this Roll number does not
Exist.')
                              elif Login_mode02 == "3":
                                   break
                              else:
                                   print("Invalid Input")

                    elif Login_mode01 == "3":
                         break
                    else:
                        print("IIError: Invalid Input")

          elif Log == "2":
               quit()
          else:
               print("IIError: Invalid Input")
```