|  |
| --- |
| **<<SCHOOL MANAGEMENT>>**  **21CSS101J – PROGRAMMING FOR PROBLEM-SOLVING**  **Mini Project Report**  *Submitted by*  **SANJAY BASKAR[Reg. No.: RA2311031010123]**  **B.Tech. CSE - <<IT>>**  **HARISH KARTHICK[Reg. No.: RA2311031010096]**  **B.Tech. CSE - <<IT>>**  **SRMIST-01.jpg**  **SCHOOL OF COMPUTING**  **COLLEGE OF ENGINEERING AND TECHNOLOGY**  **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**  **(Under Section 3 of UGC Act, 1956)**  S.R.M. NAGAR, KATTANKULATHUR – 603 203  CHENGALPATTU DISTRICT  **November 2023**  **COLLEGE OF ENGINEERING AND TECHNOLOGY**  **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**  **(Under Section 3 of UGC Act, 1956)**  S.R.M. NAGAR, KATTANKULATHUR – 603 203  **SRMIST-01.jpg**  **BONAFIDE CERTIFICATE**  Certified that Mini project report titled \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ is the bonafide work of Reg.No\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Name\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_who carried out the minor project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.  **SIGNATURE SIGNATURE**  **(GUIDE) (HEAD OF THE DEPARTMENT)** |

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S No.** | **Title** | **Page No.** |
| 1 | Problem Statement |  |
| 2 | Methodology / Procedure/ Algorithm |  |
| 3 | Flowchart |  |
| 4 | Coding (C/Python) |  |
| 5 | Front-end code (HTML, CSS, Javascript) [Optional] |  |
| 6 | Modules of the proposed work |  |
| 7 | Results/Screenshots |  |
| 8 | Conclusion |  |
| 9 | References |  |

1. **Problem Statement**

Design and implement a Python-based student and staff management system for a school. The system should facilitate the following functionalities:

Staff Management:

Staff members can sign up with a unique identifier, name, and password.

Existing staff can sign in using their credentials.

Staff members can create exams by adding questions with multiple-choice options and correct answers.

Staff can approve student accounts.

Student Management:

Students can sign up with a unique identifier, name, and password.

Existing students can sign in using their credentials.

Students can attend exams by answering multiple-choice questions.

Approved students can view their exam results.

Exam Evaluation:

Staff can approve students, allowing them to participate in exams.

Staff can publish reports, indicating whether students have passed or failed based on their exam scores.

Results and Reporting:

Students can view their exam results after evaluation.

Staff can view a detailed list of students, their approval status, exam scores, and results.

Data Persistence:

The system should store relevant data, such as staff details, student details, exam questions, and results, in a file for future reference.

User Interface:

Implement a user-friendly interface for staff and students with clear menu options and prompts.

Security Considerations:

Implement basic security measures, such as password protection.

Code Efficiency and Readability:

Ensure the code is well-organized, efficient, and follows best practices for readability.

2.**Methodology / Procedure/ Algorithm**

Certainly! Here's the algorithmic representation of the provided Python code for the sch class:

1. \*\*Class sch:\*\*

- Initialize class variables: que, i, studet, stflo, and sfid.

2. \*\*Method creatEx:\*\*

- Input the number of questions to add (n).

- Loop n times:

- Input a question (ke).

- Loop three times to input options (op).

- Input the correct answer.

- Add the question and options to the que dictionary.

3. \*\*Method stafflog:\*\*

- Display staff login options in a loop:

- If option is "1":

- Input staff details (name and password).

- Generate a staff ID (sfid).

- Add staff details to stflo dictionary.

- If option is "2":

- Try to:

- Input staff ID (st).

- Input staff password (pa).

- If ID and password match, display "Login Successfully" and call staff method.

- If not, display "User ID or Password is Wrong".

- Except for invalid ID, display "Invalid ID".

- If option is "3", exit the loop.

4. \*\*Method staff:\*\*

- Display staff options in a loop:

- If option is "1", call creatEx to create an exam.

- If option is "2":

- Input student ID for approval (aid).

- Try to:

- If student ID matches, set approval status and display approval message.

- If not, display "Invalid ID".

- Except for invalid ID, display "Invalid ID".

- If option is "3":

- Loop through students:

- If conditions met, set evaluation status to "PASS" or "FAIL".

- If no exams conducted, display "Exam Not Content".

- If option is "4":

- Display student details in a formatted table.

- Write details to a file "studentsdetails.txt".

- If option is "5", exit the loop.

5. \*\*Method stu:\*\*

- Display student login options in a loop:

- If option is "1":

- Input student details (name and password).

- Generate a student ID (i).

- Add student details to student dictionary.

- If option is "2":

- Try to:

- Input student ID (id).

- Input student password (pas).

- If ID and password match, and student is approved, call atten method.

- If not, display "User ID or Password is Wrong" or "Your not Approved".

- Except for invalid ID, display "Invalid ID".

- If option is "3", exit the loop.

6. \*\*Method atten:\*\*

- Display student options in a loop:

- If option is "1":

- If student hasn't attended, start the exam, update status, and calculate the score.

- If already attended, display "Your Already Attend Exam".

- If option is "2", display the evaluation status.

- If option is "3", exit the loop.

7. \*Instantiate Class:\*

- Create an instance of the sch class (objsch).

8. \*Main Loop:\*

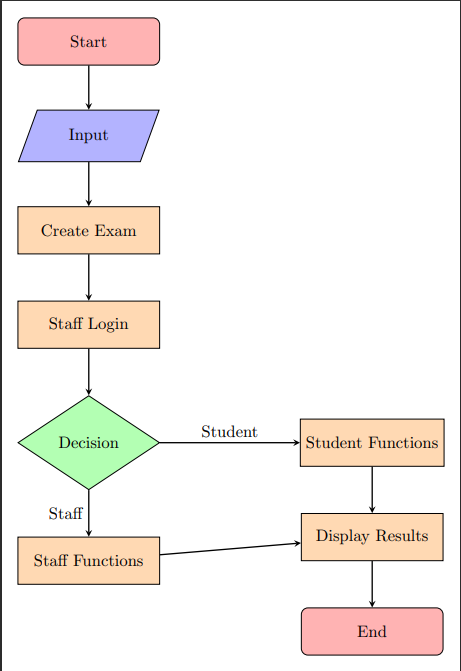
- Display main options in a loop:

- If option is "1", call stafflog.

- If option is "2", call stu.

- If option is "3", exit the program.

**3.Flow chart**

****

**4. Coding (C/Python)**

class sch:

que = {}

i = 1000

studet = {}

stflo = {}

sfid = 2000

def creatEx(self):

n = int(input("How many Question To Add:"))

for i in range(n):

ke = input("Enter the Question\n")

print("Enter the 3 options\n")

op = []

for j in range(3):

print("Enter the Option:", j + 1, end=":")

op.append(input())

op.append(input("\nEnter the Answer"))

self.que[ke] = op

def stafflog(self):

while True:

print("\n\t\t\tSTAFFLOGIN\n\t\t1.signup\n\t\t2.signin\n\t\t3.exit")

ch = input("Enter Your Option")

if ch == "1":

l = []

sfname = input("Enter Your Name:")

sfpas = input("Enter Your Password")

l.append(self.sfid)

l.append(sfname)

l.append(sfpas)

self.stflo[self.sfid] = l

print("Your staff Id Is:", self.sfid)

self.sfid += 1

print("\t\t\tSign-Up Successfully")

elif ch == "2":

try:

st = int(input("Enter Your Id:"))

pa = input("Enter Your Password:")

if st == self.stflo[st][0] and self.stflo[st][2] == pa:

print("\t\t\tLogin Successfully.....")

self.staff()

else:

print("User Id Or PassWord is Wrong")

except:

print("Invalid Id")

elif ch == "3":

break

else:

print("enter Correct Option")

def staff(self):

while True:

print("\t\t1.CreateExam\n\t\t2.Approve students\n\t\t3.publish report\n\t\t4.ViewStudents\n\t\t5.exit")

ch = input("\nEnter the option")

if ch == "1":

self.creatEx()

elif ch == "2":

aid = int(input("Enter the Id for Approve:"))

try:

if aid == self.studet[aid][0]:

self.studet[aid][3] = True

self.studet[aid][6] = "Approved"

print(aid, "Is Approved")

except:

print("Invalid Id")

elif ch == "3":

f = True

for i in self.studet:

f = False

if len(self.studet) / 2 <= self.studet[i][5] and self.studet[i][7] == "Not Evaluated":

self.studet[i][7] = "PASS"

elif self.studet[i][7] == "Not Evaluated":

self.studet[i][7] = "FAIL"

if f:

print("Exam Not Contect")

elif ch == "4":

print("Student Id StudentName Approved or Not Score Results")

print("========== =========== =============== ===== ==========")

with open("studentsdetails.txt", 'w') as f1:

print("Student Id StudentName Approved or Not Score Results", file=f1)

for i in self.studet:

print(

"%-10s %-11s %-14s %-5d %-7s" % (

i, self.studet[i][1], self.studet[i][6], self.studet[i][5], self.studet[i][7]))

print("-----------------------------------------------------------------------")

with open("studentsdetails.txt", 'a') as f:

print(

"%-10s %-11s %-14s %-5d %-7s" % (

i, self.studet[i][1], self.studet[i][6], self.studet[i][5], self.studet[i][7]), file=f)

print("=======================================================================")

elif ch == "5":

break

def stu(self):

while True:

print("\n\t\t\tSTUDENTLOGIN\n\t\t1.signup\n\t\t2.signin\n\t\t3.exit")

ch = input("enter the Option:")

if ch == "1":

l = []

stnam = input("Enter your name:")

stpas = input("Enter Your Password:")

l.append(self.i)

l.append(stnam)

l.append(stpas)

l.append(False)

l.append(False)

l.append(0)

l.append("Not Approved")

l.append("Not attend")

self.studet[self.i] = l

print("Your Id is:", self.studet[self.i][0])

self.i += 1

elif ch == "2":

id = int(input("Enter Your Id:"))

pas = input("Enter Your PassWord:")

try:

if self.studet[id][2] == pas and self.studet[id][0] == id:

if True == self.studet[id][3]:

print("\t\t\tLogin Successfully...")

self.atten(id)

self.studet[id][4] = True

else:

print("Your not Approved")

else:

print("UserID or PassWord Is Wrong")

except:

print("Invalid Id")

elif ch == "3":

break

def atten(self, id):

while True:

print("\n\t\t1.AttendExam\n\t\t2.Result\n\t\t3.exit")

chs = input("Enter Your Option")

if chs == "1":

if self.studet[id][7] == "Not attend":

c = 0

print("\n\t\t\ttest started.....\n")

self.studet[id][7] = "Not Evaluated"

for i in self.que:

print(i)

print("1:", self.que[i][0], "2:", self.que[i][1], "3:", self.que[i][2])

ans = input("Enter Your Answer:")

if self.que[i][3] == ans:

c += 1

self.studet[id][5] = c

else:

print("Your Already AttendExam")

elif chs == "2":

print(self.studet[id][7])

elif chs == "3":

break

objsch = sch()

print("\t\t\tTHE ")

while True:

print("\t\t1.Staff\n\t\t2.student\n\t\t3.exit")

n = input("enter the input:")

if n == '1':

objsch.stafflog()

elif n == "2":

objsch.stu()

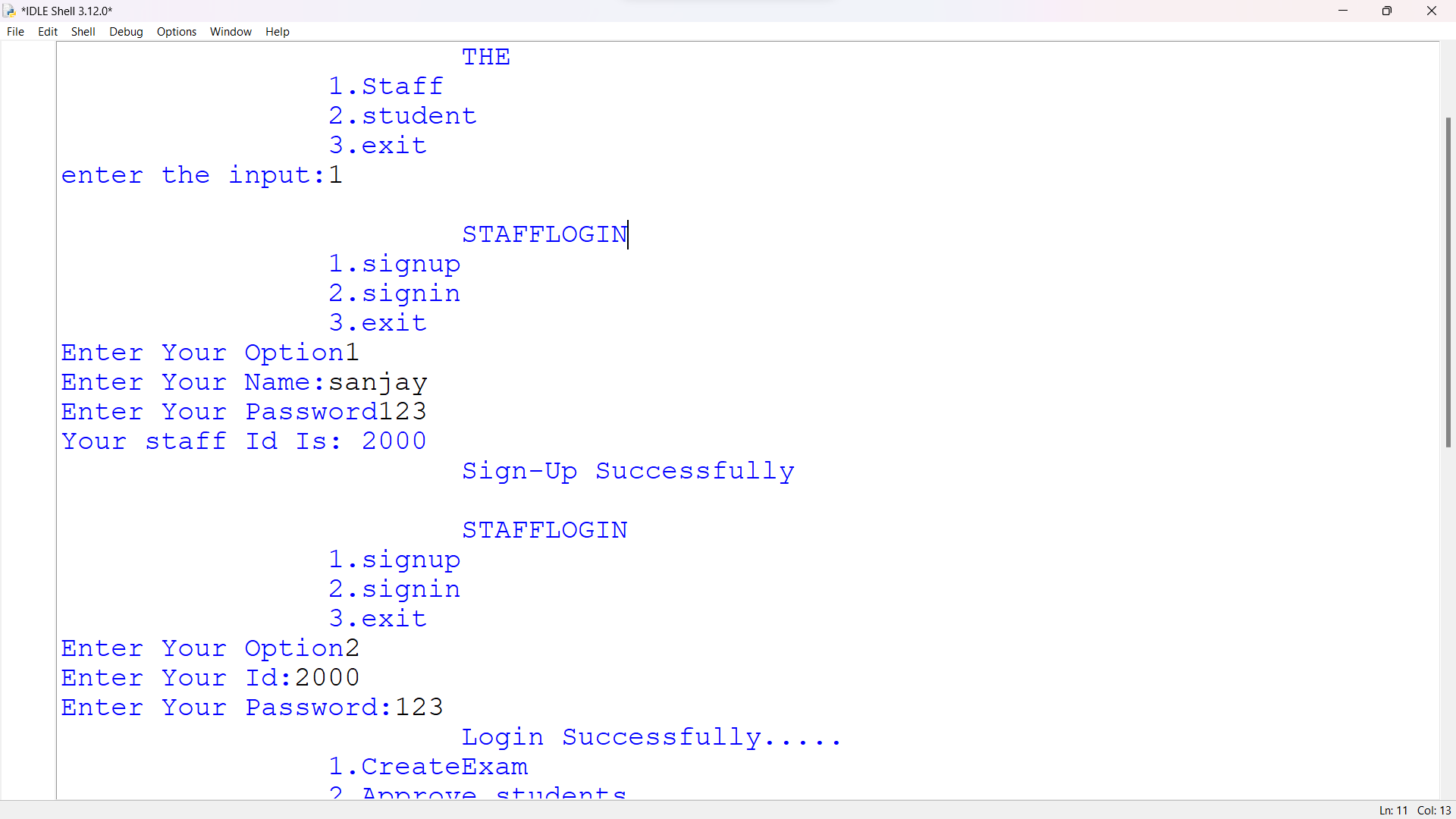
elif n == "3":

break

**6. Modules of the proposed work**

* class Sch
* def create(self);
* def Stafflog(self);
* def Staff(self);
* def Stu(self);
* def after(self,id);

**7. Results/Screenshots**

****

**A screenshot of a computer program

Description automatically generated**

**8. Conclusion**

The provided Python script represents a basic implementation of a school management system. Here's a conclusion summarizing the key aspects of the work:

Functionality:

The program allows staff to create exams, approve students, publish reports, and view student details.

Students can sign up, log in, attend exams, and view their results.

Modularity:

The code is organized into different modules, each handling specific functionalities.

Modules include exam creation, staff and student authentication, staff actions, student attendance, main program execution, data storage, and file handling.

Data Structures:

Dictionaries are used for storing information about questions (que), staff (stflo), and students (studet).

User Interaction:

The program interacts with users through a series of input prompts and menu options.

Staff and students have distinct functionalities and actions.

File Handling:

Student details are written to a file named "studentsdetails.txt".

Improvements:

The code could benefit from error handling and validation to enhance robustness.

Additional features, such as more comprehensive reporting, could be added to improve the system's capabilities.

Code Style:

The code follows a reasonable indentation and structure, making it readable and understandable.

Areas for Enhancement:

The program could be extended to include more features like grading, course management, and a more sophisticated user interface.

Error handling and input validation could be improved to enhance the program's reliability.

In conclusion, the provided script is a basic implementation of a school management system with room for enhancements and additional features. It serves as a foundation that can be expanded and improved based on specific requirements and use cases

**9. References**

**Python Documentation:**

**Official Python Documentation: The Python documentation is an excellent resource for understanding the Python language itself, including its syntax, built-in functions, and standard libraries.**

**Object-Oriented Programming (OOP):**

**Real Python - Object-Oriented Programming in Python: This tutorial provides a comprehensive introduction to OOP in Python.**

**File Handling in Python:**

**Real Python - Reading and Writing Files in Python: Understanding file handling is crucial, especially when dealing with storing and retrieving data.**

**Error Handling and Exception Handling:**

**Real Python - Python Exception Handling: Learn about handling errors and exceptions in Python to make your code more robust.**

**Python Modules and Packages:**

**Real Python - Python Modules and Packages: Explore how to organize your code into modules and packages for better maintainability.**

**Enhancing User Interface:**

**If you want to create a graphical user interface (GUI), you can explore libraries like Tkinter, PyQt, or Kivy.**