JUZZ-IT EDUCATION

ADVANCED E-LEARNING PLATFORM

GROUP - A PROJECT

N M Suhas

Rinesh A R

Induvathi G

Praveen S

Sanjay S

AGENDA

Project Name : Student Management System

1. Introduction
2. Abstract
3. Languages used
4. Platform used
5. ER Diagram
6. Use case diagram
7. Coding
8. Output
9. Conclusion

INTRODUCTION TO SMS

- Student Management System can handle all the details about a student. The details include student's personal details, academic details etc..
- The student management system is an automated version of manual Student Management System.
- A student management system is designed to record, analyze, and manage information in a schools or College.

ABSTRACT

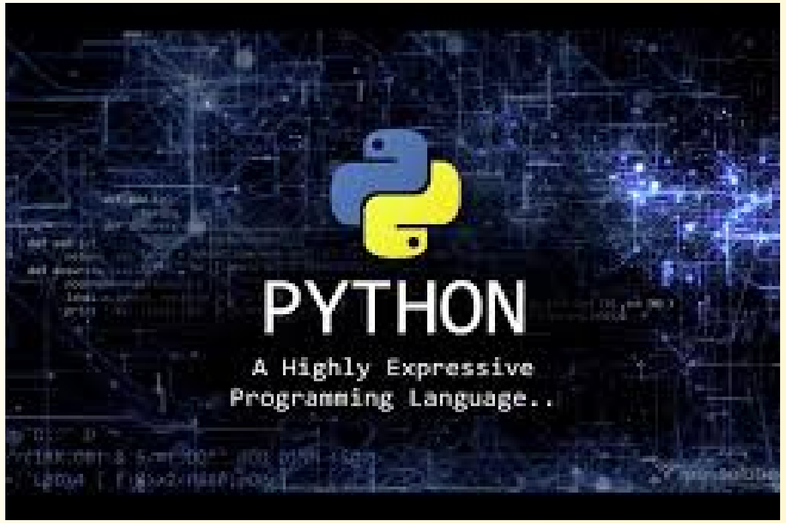
The Student Management System (SMS) is a comprehensive software solution designed to streamline and enhance the management of student-related processes within educational institutions.

Key Features under Student and Teacher Table

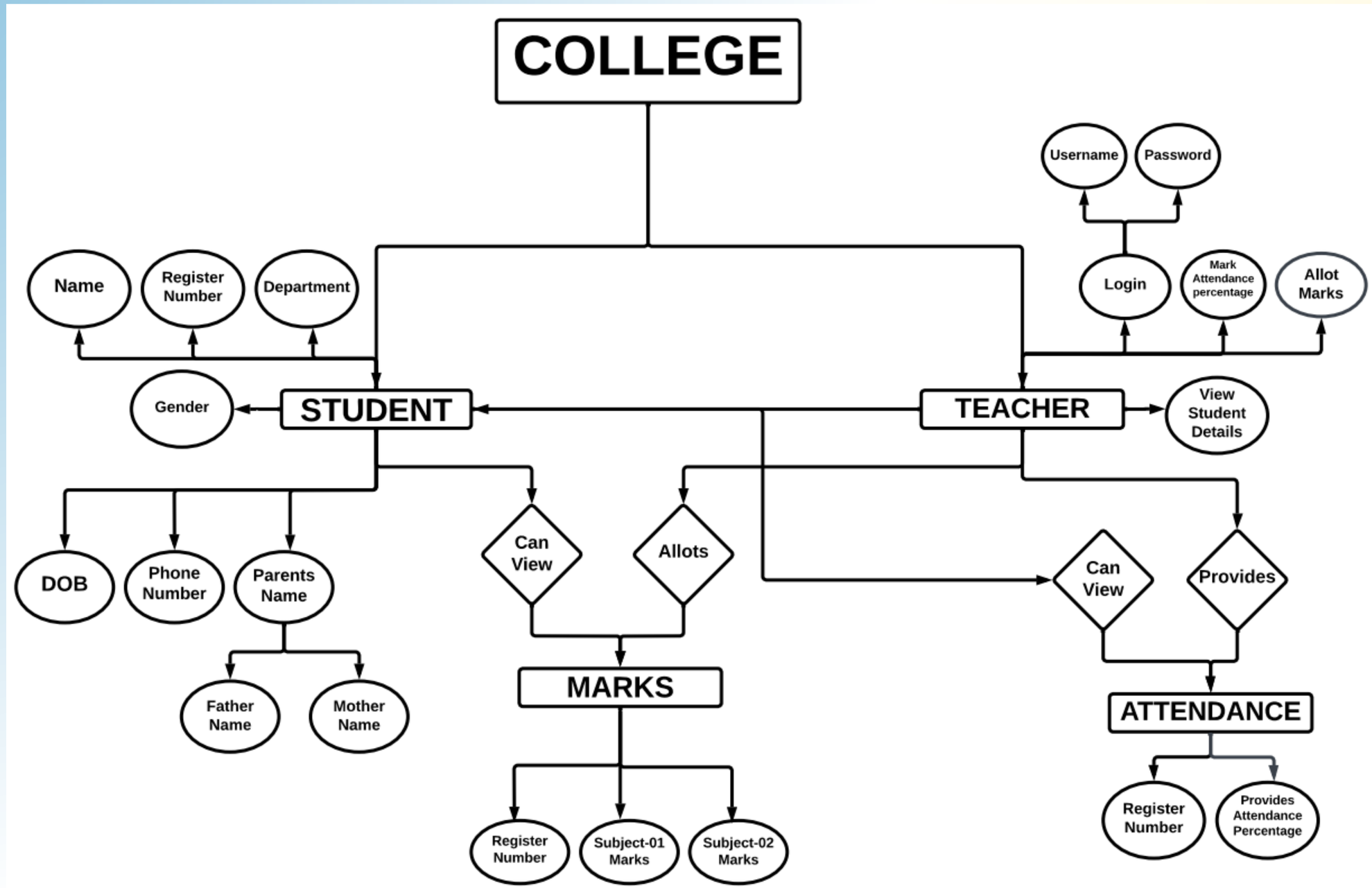
- Data Acceptance
- Data Display
- Data Search
- Data Deletion
- Data Update
- View Results
- View Attendance
- Login
- Provide Attendance
- Allot Marks
- View Student Details

Languages & Platform used to code

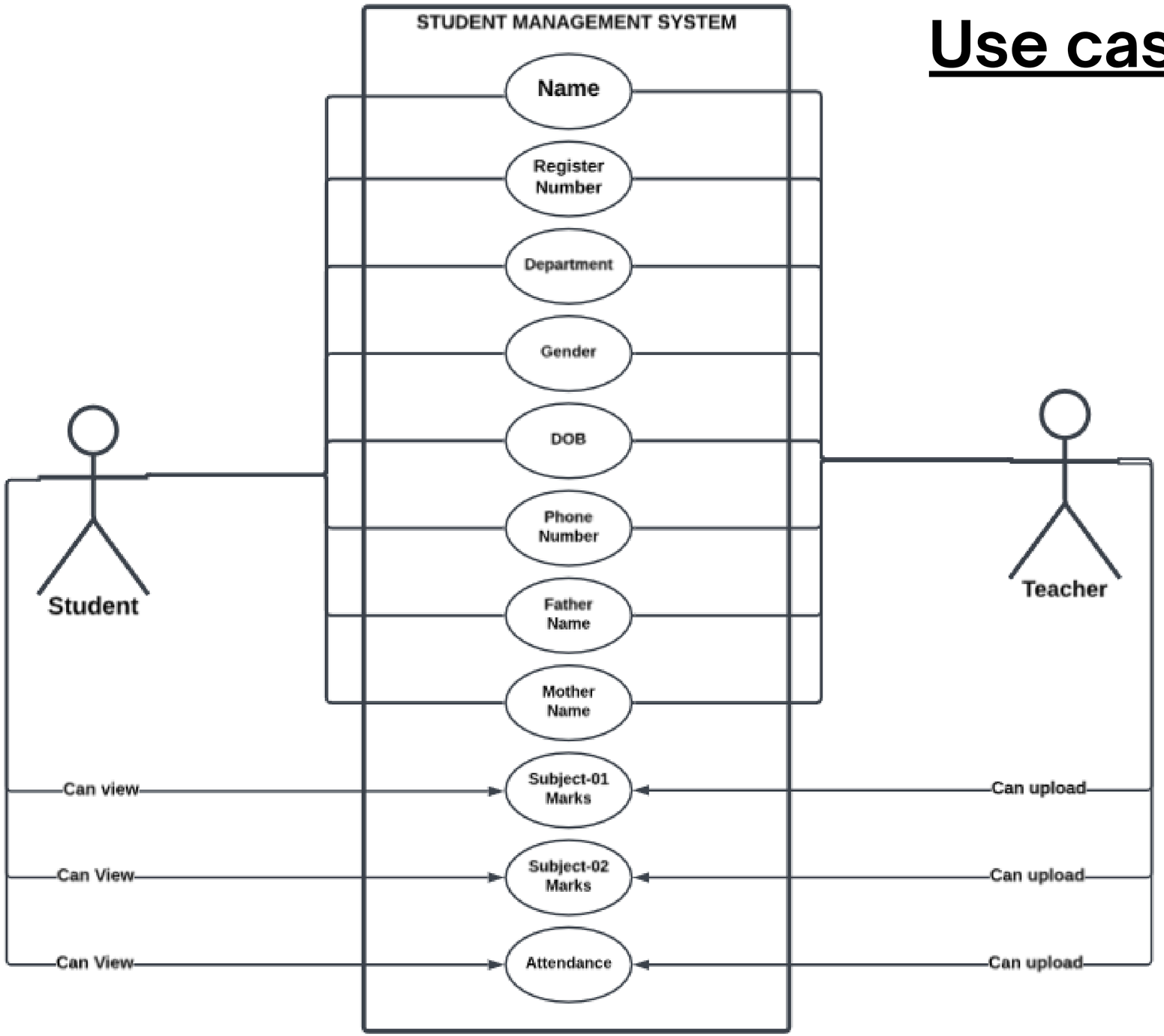
Google Colab is a cloud-based platform provided by Google that allows you to write and execute Python code in a collaborative environment. It's popular among data scientists and machine learning because it provides free access to GPU and TPU resources.



ER DIAGRAM FOR STUDENT MANAGEMENT



Use case diagram



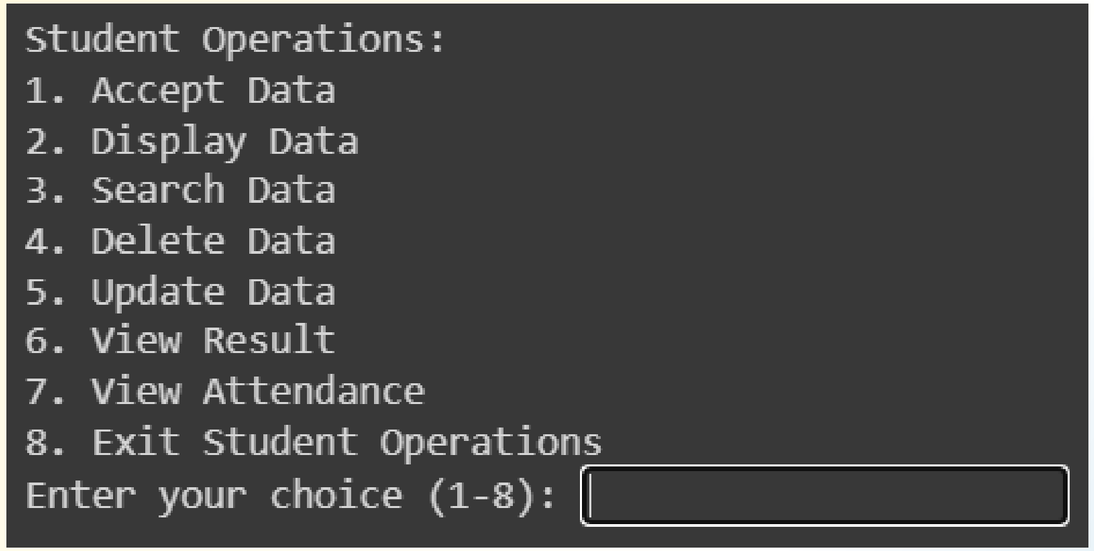
CODING

```
class StudentManagementSystem:
  def __init__(self):
    self.students = {}
    self.teachers = {'vishal': 'vishal123'}
    self.logged_in_user = None

  def login(self, username, password):
    if username in self.teachers and self.teachers[username] == password:
      print(f"Welcome, Teacher {username}!")
      self.logged_in_user = 'teacher'
    else:
      print("Invalid username or password. Please try again.")
```

```
Student Management System Menu:
1. Student
2. Teacher
3. Exit
Enter your choice (1-3): 2
Enter teacher username: vishal
Enter teacher password: vishal123
Welcome, Teacher vishal!
```

STUDENT OPERATION

```
def student_operations(self):
    while True:
        print("\nStudent Operations:")
        print("1. Accept Data")
        print("2. Display Data")
        print("3. Search Data")
        print("4. Delete Data")
        print("5. Update Data")
        print("6. View Result")
        print("7. View Attendance")
        print("8. Exit Student Operations")

        student_choice = input("Enter your choice (1-8): ")
```




```
        if student_choice == '1':
            self.accept_data()
        elif student_choice == '2':
            self.display_data()
        elif student_choice == '3':
            roll_number = input("Enter roll number to search: ")
            self.search_data(roll_number)
        elif student_choice == '4':
            roll_number = input("Enter roll number to delete: ")
            self.delete_data(roll_number)
        elif student_choice == '5':
            roll_number = input("Enter roll number to update: ")
            self.update_data(roll_number)
        elif student_choice == '6':
            roll_number = input("Enter your roll number to view result: ")
            self.logged_in_user = roll_number
            self.view_result(roll_number)
        elif student_choice == '7':
            roll_number = input("Enter your roll number to view attendance: ")
            self.logged_in_user = roll_number
            self.view_attendance(roll_number)
        elif student_choice == '8':
            break
        else:
            print("Invalid choice. Please enter a valid option (1-8).")
```

TEACHER OPERATION

```
def teacher_operations(self):
    while True:
      print("\nTeacher Operations:")
      print("1. Provide Attendance")
      print("2. Allot Marks for 2 Subjects")
      print("3. View Student Details")
      print("4. Exit Teacher Operations")

      teacher_choice = input("Enter your choice (1-4): ")

      if teacher_choice == '1':
        if self.logged_in_user == 'teacher':
          roll_number = input("Enter student roll number to provide attendance: ")
          self.provide_attendance(roll_number)
        else:
          print("Permission denied. Please login as a teacher.")
      elif teacher_choice == '2':
        if self.logged_in_user == 'teacher':
          roll_number = input("Enter student roll number to allot marks: ")
          self.allot_marks(roll_number)
        else:
          print("Permission denied. Please login as a teacher.")
      elif teacher_choice == '3':
        if self.logged_in_user == 'teacher':
          roll_number = input("Enter student roll number to view details: ")
          self.view_student_details(roll_number)
        else:
          print("Permission denied. Please login as a teacher.")
      elif teacher_choice == '4':
          break
      else:
      print("Invalid choice. Please enter a valid option (1-4).")
```

```
Teacher Operations:
1. Provide Attendance
2. Allot Marks for 2 Subjects
3. View Student Details
4. Exit Teacher Operations
Enter your choice (1-4):
```

ACCEPT OPERATION

```
def accept_data(self):
    name = input("Enter student name: ")
    roll_number = input("Enter roll number: ")
    department = input("Enter department (CSE/ME/EEE): ")
    gender = input("Enter Gender: ")
    dob = input("Enter Date of Birth (DD-MM-YYYY): ")
    phone_number = input("Enter phone number: ")
    father_name = input("Enter Father's name(As per 10th marksheet): ")
    mother_name = input("Enter Mother's name(As per 10th marksheet): ")

    self.students[roll_number] = {
      'name': name,
      'Registration number': roll_number,
      'department': department,
      'gender': gender,
      'dob': dob,
      'phone number': phone_number,
      'father name': father_name,
      'mother name': mother_name,
      'result': None
    }
    print(f"Data for {name} (Roll No: {roll_number}) added successfully.")
```

```
Enter your choice (1-8): 1
Enter student name: N M SUHAS
Enter roll number: 408CS21029
Enter department (CSE/ME/EEE): CSE
Enter Gender: Male
Enter Date of Birth (DD-MM-YYYY): 26-06-2005
Enter phone number: 7892331225
Enter Father's name(As per 10th marksheet): N B MOHAN
Enter Mother's name(As per 10th marksheet): N M VINUTHA
Data for N M SUHAS (Roll No: 408CS21029) added successfully.
```

DISPLAY OPERATION

```
def display_data(self):
    for roll_number, data in self.students.items():
      print(f"\nStudent Details for Roll No {roll_number}:")
      for key, value in data.items():
        if key not in ('result', 'attendance'):
          print(f"{key}: {value}")
```

```
Enter your choice (1-8): 2

Student Details for Roll No 408CS21029:
name: N M SUHAS
Registration number: 408CS21029
department: CSE
gender: Male
dob: 26-06-2005
phone number: 7892331225
father name: N B MOHAN
mother name: N M VINUTHA
```

SEARCH OPERATION

```
def search_data(self, roll_number):
    if roll_number in self.students:
      print(f"\nStudent Details for Roll No {roll_number}:")
      for key, value in self.students[roll_number].items():
        print(f"{key}: {value}")
    else:
      print(f"Student with Roll No {roll_number} not found.")
```

```
Enter your choice (1-8): 3
Enter roll number to search: 408CS21029

Student Details for Roll No 408CS21029:
name: N M SUHAS
Registration number: 408CS21029
department: CSE
gender: Male
dob: 26-06-2005
phone number: 7892331225
father name: N B MOHAN
mother name: N M VINUTHA
result: None
```

UPDATE OPERATION

```
def update_data(self, roll_number):
    if roll_number in self.students:
      print(f"\nUpdate Details for Roll No {roll_number}:")
      name = input("Enter updated name: ")
      department = input("Enter updated department (CSE/ME/EEE): ")
      gender = input("Enter Gender: ")
      dob = input("Enter updated Date of Birth (DD-MM-YYYY): ")
      phone_number = input("Enter new phone number: ")
      father_name = input("Enter updated Father's name: ")
      mother_name = input("Enter updatedMother's name: ")

      self.students[roll_number]['name'] = name
      self.students[roll_number]['department'] = department
      self.students[roll_number]['Gender'] = gender
      self.students[roll_number]['DOB'] = dob
      self.students[roll_number]['phone number'] = phone_number
      self.students[roll_number]['father name'] = father_name
      self.students[roll_number]['mother name'] = mother_name

      print(f"Record for Roll No {roll_number} updated successfully.")
    else:
      print(f"Student with Roll No {roll_number} not found.")
```

```
Enter your choice (1-8): 5
Enter roll number to update: 408CS21029

Update Details for Roll No 408CS21029:
Enter updated name: SUHAS N M
Enter updated department (CSE/ME/EEE): MECHANICAL
Enter Gender: Male
Enter updated Date of Birth (YYYY-MM-DD): 27-06-2005
Enter new phone number: 1234567891
Enter updated Father's name: MOHAN N B
Enter updatedMother's name: VINUTHA N M
Record for Roll No 408CS21029 updated successfully.
```

```
Enter your choice (1-8): 2

Student Details for Roll No 408CS21029:
name: SUHAS N M
Registration number: 408CS21029
department: MECHANICAL
gender: Male
dob: 26-06-2005
phone number: 1234567891
father name: MOHAN N B
mother name: VINUTHA N M
subject1_marks: 82.0
subject2_marks: 92.0
Gender: Male
DOB: 27-06-2005
```

DELETE OPERATION

```
def delete_data(self, roll_number):
    if roll_number in self.students:
      del self.students[roll_number]
      print(f"Record for Roll No {roll_number} deleted successfully.")
    else:
      print(f"Student with Roll No {roll_number} not found.")
```

```
Student Operations:
1. Accept Data
2. Display Data
3. Search Data
4. Delete Data
5. Update Data
6. View Result
7. View Attendance
8. Exit Student Operations
Enter your choice (1-8): 4
Enter roll number to delete: 408CS21029
Record for Roll No 408CS21029 deleted successfully.
```

PROVIDE ATTENDANCE

```
def provide_attendance(self, roll_number):
    if roll_number in self.students:
      attendance_percentage = float(input("Enter attendance percentage for the student: "))
      self.students[roll_number]['attendance'] = attendance_percentage
      print(f"Attendance filled successfully for Roll No {roll_number}.")
    else:
      print(f"Student with Roll No {roll_number} not found.")
```

```
Teacher Operations:
1. Provide Attendance
2. Allot Marks for 2 Subjects
3. View Student Details
4. Exit Teacher Operations
Enter your choice (1-4): 1
Enter student roll number to provide attendance: 408CS21029
Enter attendance percentage for the student: 82
Attendance filled successfully for Roll No 408CS21029.
```

ALLOT MARKS FOR STUDENTS

```
def allot_marks(self, roll_number):
    if roll_number in self.students:
      subject1_marks = float(input("Enter marks for Subject 1: "))
      subject2_marks = float(input("Enter marks for Subject 2: "))
      self.students[roll_number]['subject1_marks'] = subject1_marks
      self.students[roll_number]['subject2_marks'] = subject2_marks

      print(f"Marks assigned successfully for Roll No {roll_number}.")
    else:
      print(f"Student with Roll No {roll_number} not found.")
```

```
Enter your choice (1-4): 2
Enter student roll number to allot marks: 408CS21029
Enter marks for Subject 1: 86
Enter marks for Subject 2: 92
Marks assigned successfully for Roll No 408CS21029.
```

VIEW STUDENT DETAILS IN TEACHER TABLE

```
def view_student_details(self, roll_number):
    if roll_number in self.students:
      print(f"\nStudent Details for Roll No {roll_number}:")
      for key, value in self.students[roll_number].items():
        print(f"{key}: {value}")
      self.view_result(roll_number)
    else:
      print(f"Student with Roll No {roll_number} not found.")
```

```
Enter your choice (1-4): 3
Enter student roll number to view details: 408CS21029

Student Details for Roll No 408CS21029:
name: N M SUHAS
Registration number: 408CS21029
department: CSE
gender: Male
dob: 26-06-2005
phone number: 7892331225
father name: N B MOHAN
mother name: N M VINUTHA
result: None
attendance: 82.0
subject1_marks: 86.0
subject2_marks: 92.0
Gender: Male
DOB: 26-06-2005

Result for Roll No 408CS21029:
Subject 1 Marks: 86.0
Subject 2 Marks: 92.0
Total Marks: 178.0
```

VIEW RESULT IN STUDENT TABLE

```
def view_result(self, roll_number):
    if roll_number in self.students:
     # Check if the logged-in user is a student and matches the provided roll_number
     if self.logged_in_user == roll_number:
        if 'subject1_marks' in self.students[roll_number] and 'subject2_marks' in self.students[roll_number]:
          subject1_marks = self.students[roll_number]['subject1_marks']
          subject2_marks = self.students[roll_number]['subject2_marks']
          total_marks = subject1_marks + subject2_marks
          print(f"\nResult for Your Roll No {roll_number}:")
          print(f"Subject 1 Marks: {subject1_marks}")
          print(f"Subject 2 Marks: {subject2_marks}")
          print(f"Total Marks: {total_marks}")
        else:
          print("Marks not available. Please ask the teacher to allot marks.")
```

VIEW RESULT IN STUDENT TABLE

Continuation

```
elif self.logged_in_user == 'teacher':
    if 'subject1_marks' in self.students[roll_number] and 'subject2_marks' in self.students[roll_number]:
        subject1_marks = self.students[roll_number]['subject1_marks']
        subject2_marks = self.students[roll_number]['subject2_marks']
        total_marks = subject1_marks + subject2_marks
        print(f"\nResult for Roll No {roll_number}:")
        print(f"Subject 1 Marks: {subject1_marks}")
        print(f"Subject 2 Marks: {subject2_marks}")
        print(f"Total Marks: {total_marks}")
    else:
        print("Marks not available. Please allot marks for the student.")
  else:
    print("Permission denied. You can only view your own result.")
 else:
  print(f"Student with Roll No {roll_number} not found.")
```

```
Enter your choice (1-8): 6
Enter your roll number to view result: 408CS21029

Result for Your Roll No 408CS21029:
Subject 1 Marks: 86.0
Subject 2 Marks: 92.0
Total Marks: 178.0
```

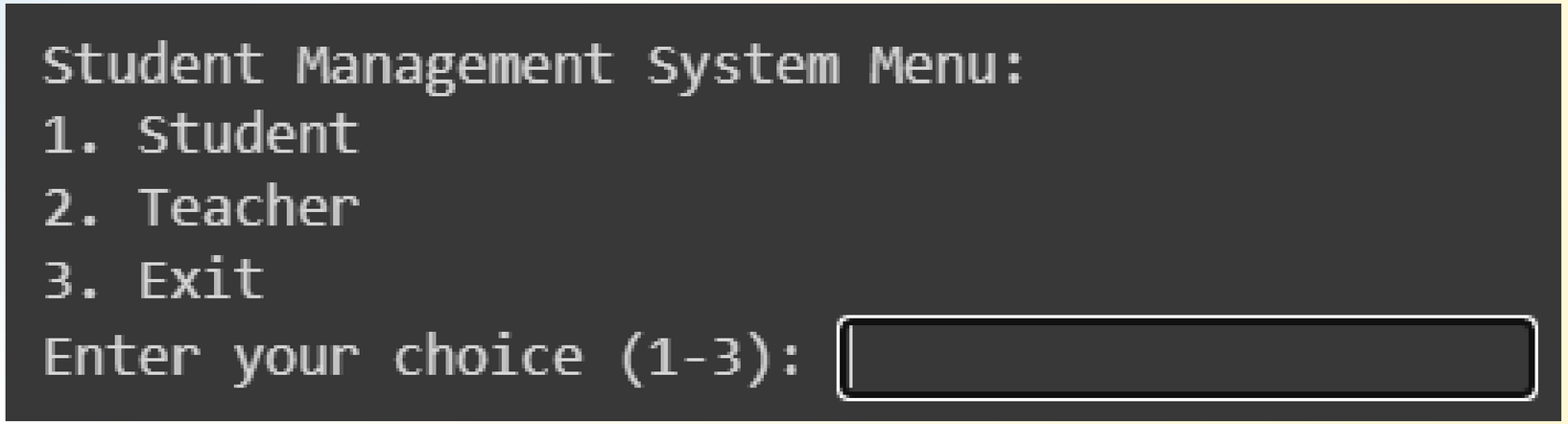
VIEW ATTENDANCE IN STUDENT TABLE

```
def view_attendance(self, roll_number):
    if roll_number in self.students:
        # Check if the logged-in user is a student and matches the provided roll_number
        if self.logged_in_user == roll_number:
            if 'attendance' in self.students[roll_number]:
                attendance_percentage = self.students[roll_number]['attendance']
                print(f"\nAttendance for Your Roll No {roll_number}:")
                print(f"Attendance Percentage: {attendance_percentage}%")
            else:
                print("Attendance not available. Please ask the teacher to provide attendance.")
        else:
            print("Permission denied. You can only view your own attendance.")
    else:
        print(f"Student with Roll No {roll_number} not found.")
```

```
Enter your choice (1-8): 7
Enter your roll number to view attendance: 408CS21029

Attendance for Your Roll No 408CS21029:
Attendance Percentage: 82.0%
```

DEMONSTRATION OF THE CODE

```
sms = StudentManagementSystem()

while True:
  print("\nStudent Management System Menu:")
  print("1. Student")
  print("2. Teacher")
  print("3. Exit")
```



CONTINUATION CODE

```
choice = input("Enter your choice (1-3): ")

 if choice == '1':
 # Set the logged-in user as a student when entering the student menu
 sms.logged_in_user = 'student'
 sms.student_operations()
 elif choice == '2':
 username = input("Enter teacher username: ")
 password = input("Enter teacher password: ")
 sms.login(username, password)
 if sms.logged_in_user == 'teacher':
 sms.teacher_operations()
 elif choice == '3':
 print("Exiting Student Management System. Goodbye!")
 break
 else:
 print("Invalid choice. Please enter a valid option (1-3).")
```

CONCLUSION

1. **Structured System:** The StudentManagementSystem class offers a well-organized and interactive framework for managing student information.
2. **Comprehensive Operations:** The class incorporates methods that handle a diverse set of operations, including accepting, displaying, searching, updating, and deleting student data.
3. **Data Acceptance:** The accept_data method facilitates the input of detailed student information, encompassing aspects such as name, roll number, age, gender, etc...
4. **Data Display:** The display_data method provides a comprehensive overview of all stored student records.
5. **Search Functionality:** The search_data method allows for efficient searching, supporting queries based on either the full roll number or the last two digits of the roll number.
6. **Data Modification:** The update_data method enables the modification of a student's name and marks, utilizing the provided roll number.
7. **Record Deletion:** The delete_data method removes a student's record from the system based on the specified roll number.
8. **User-Friendly Interface:** Overall, the **StudentManagementSystem** class provides a flexible and user-friendly interface tailored for both students and teachers.
9. **Efficient Management:** The integrated functionalities contribute to efficient management and retrieval of student information.

REFERENCES

CLICK HERE FOR COLAB CODE LINK

CLICK HERE FOR ER-DIAGRAM LINK

CLICK HERE FOR USE CASE LINK

THANK YOU