**Scenario Overview:**

You are tasked with developing a **Student Management System** for a university or school. This system will serve as the core platform to manage various academic functions, such as enrolling students in courses, assigning teachers to courses, tracking student performance, and managing departments. The goal is to create a scalable, efficient, and maintainable system by leveraging **Object-Oriented Programming (OOP)** principles.

The system will be used by different stakeholders, including:

• **Students**: To view their courses, grades, and attendance.

• **Teachers**: To manage their courses, view students, and update grades and attendance.

• **Administrators/Department Heads**: To oversee departments, assign teachers to courses, and monitor overall student performance.

**Features of the System:**

1. **Student Management**: Students can be enrolled in multiple courses, have their grades tracked, and have attendance monitored. Each student’s GPA is automatically calculated based on the grades they receive.

2. **Course Management**: Courses have unique course codes and are assigned to teachers. Each course tracks the students enrolled in it along with their grades and attendance.

3. **Teacher Management**: Teachers are responsible for one or more courses. They can view their students, update their grades, and monitor attendance. Teachers are associated with specific departments.

4. **Department Management**: Departments oversee multiple courses and teachers. They can list all courses offered, assign teachers to courses, and list all students enrolled in any department courses.

**Functionalities in Detail:**

**1. Student Enrollment:**

• Students can enroll in multiple courses. The add\_course() method in the Student class will add a course to the student’s list of enrolled courses.

• The system will ensure that no duplicate course enrollments occur.

**2. Grade and Attendance Management:**

• Teachers can update grades and attendance for each student. This will be done through the update\_grade() and update\_attendance() methods.

• When grades are updated, the student’s GPA is recalculated automatically via the calculate\_gpa() method.

**3. Teacher Management:**

• Teachers are assigned to courses using the assign\_course() method in the Teacher class. They are responsible for managing grades and attendance for the students enrolled in their courses.

**4. Department Management:**

• The Department class will oversee all courses and teachers within the department. It will also allow the administrator to list all students enrolled in department courses, giving a comprehensive view of the department’s operations.

• Courses and teachers can be added to departments using the add\_course() and add\_teacher() methods.

**5. GPA Calculation:**

• Each student’s GPA is automatically calculated based on their grades in enrolled courses. The GPA is recalculated every time a grade is updated.

• The formula for GPA calculation is based on the average of the grades across all enrolled courses.

**6. Reports:**

• Administrators can generate reports that provide detailed insights into students, courses, and departments.

• Teachers can view a list of all students in their courses, along with their grades and attendance.

• Students can view their own performance across all enrolled courses.

**Stakeholders and Their Interactions with the System:**

1. **Students**:

• Enroll in multiple courses.

• Track their grades and attendance.

• View their GPA.

• Receive notifications when their grades or attendance records are updated.

2. **Teachers**:

• Teach multiple courses within a department.

• Update student grades and attendance.

• View and manage the students enrolled in their courses.

• Receive assignments from department heads regarding courses.

3. **Administrators/Department Heads**:

• Oversee multiple teachers and courses within a department.

• Assign teachers to courses.

• Track student performance across courses within the department.

• Generate reports for student performance, course enrollments, and department metrics.

**System Design**

We will use **Object-Oriented Programming (OOP)** principles to design the system. The system will be broken down into manageable classes that represent real-world entities like **students**, **teachers**, **courses**, and **departments**.