# Student Management System

A Console-based Java Project using OOP Concepts

In educational institutions, managing student data through manual or semi-digital processes often leads to inefficiencies, data redundancy, and security risks.

There is a need for a fully digital system that ensures secure, structured, and accessible handling of student registrations, courses, exams, and result processing.

The solution should support basic academic workflows while applying object-oriented programming concepts for modularity and scalability.

This project is a Console-Based Student Management System developed in Java, applying Object-Oriented Programming concepts such as Encapsulation, Abstraction, Inheritance, and Polymorphism.

Main Features:

* Student Registration
* Course Assignment
* Exam Scheduling
* Result Management
* Admin Panel with secure login

The system is structured in three logical layers:

1. User Layer:
   * Interface for admin and students via console I/O.
2. Business Logic Layer:
   * Manages core functionalities like registrations, assignments, scoring, etc.
3. Data Layer:
   * Maintains runtime in-memory data using lists and maps.

Actors:

* Admin
* Student

Use Cases:

* Admin: Login, Register Student, Add Course, Assign Course, Schedule Exam, Assign Scores, View Results
* Student: View Courses, View Results

Main Classes:

1. Student:
   * Attributes: ID, name, email, registeredCourses, examScores
   * Methods: registerCourse, viewCourses, viewResults, assignScore
2. Course:
   * Attributes: courseId, courseName, credits
3. Result:
   * Methods: calculateGrade
4. Admin:
   * Methods: login, registerStudent, addCourse, assignCourseToStudent, assignScore
5. Main (Driver):
   * Controls program flow and user interaction.

Admin Flow:

1. Login
2. Register Students
3. Add Courses
4. Assign Courses to Students
5. Assign Scores for Exams
6. View Student List and Results

Student Flow:

1. Enter Student ID
2. View Registered Courses
3. View Results

Encapsulation:

* + Keeping data private within classes and exposing necessary methods.

Abstraction:

* + Hiding complex logic and exposing functionality through method interfaces.

Inheritance (Extendable):

* + Shared attributes/methods through base classes (can be added in future).

Polymorphism:

* + Method overloading and extendable interfaces for future enhancements.

Included Files:

* + Student.java
  + Course.java
  + Result.java
  + Admin.java
  + Main.java

Code implements:

* + Registration system
  + Course and score management
  + Result calculation and viewing
  + Admin security access

Use the full source files to compile and run the system in any Java environment.