

College Management System

Project Objective

The purpose of this project is to design and implement a console-based Java *College Management System* that manages student validation, course catalog operations, seat availability checks, and course enrollment using a credit-based fee calculation structure. Students may enroll only in existing courses that still have available seats, and each enrollment must compute the applicable fee based on the number of credits attached to the chosen course.

A. System Design

| Package Name | Description |
|-----------------------|--|
| com.wipro.cms.entity | This package contains all entity classes that represent the essential academic structures of the college: Student, Course, and Enrollment. These classes store fundamental academic information using encapsulation principles. Each entity models real institutional characteristics such as course credits, student programs, and enrollment identifiers, ensuring the system mirrors realistic administrative conditions. |
| com.wipro.cms.util | This package contains all user-defined exception classes. These exceptions help maintain the integrity of academic workflows by preventing invalid operations such as enrolling nonexistent students, selecting invalid courses, or exceeding course capacity. By isolating exceptions into their own package, the system achieves clear separation of concerns and structured error handling. |
| com.wipro.cms.service | This package contains the business logic that drives the entire system. It validates student details, checks course capacity, manages enrollments and withdrawals, and calculates fees based on the academic credit structure. This layer enforces the college's operational rules and ensures only valid, logically consistent actions are processed. |
| com.wipro.cms.main | This package contains the Main class, which initializes the system using ArrayLists and demonstrates all core functionalities, including student validation, course enrollment, fee calculation, and printing student enrollment summaries. |

B. Package: com.wipro.cms.util (Exception Classes)

| Exception Class | Method / Fields | Description |
|-------------------------|--------------------------------|---|
| InvalidStudentException | public String toString() | This exception is thrown when an enrollment request references a student who does not exist in the registered student list. It ensures that only valid, authenticated students can participate in academic processes, preventing incorrect or fraudulent enrollment attempts. The exception |

| Exception Class | Method / Fields | Description |
|----------------------------|-----------------------------|--|
| | | safeguards institutional integrity by enforcing correct student identification at all times. |
| CourseFullException | public String toString() | This exception occurs when a course has reached its maximum allowed capacity. Colleges enforce strict seat limits due to infrastructure, faculty availability, or accreditation constraints. This exception prevents violation of institutional policies by blocking additional enrollments once the capacity threshold is reached. |
| EnrollmentException | public String toString() | This exception is thrown when an enrollment operation fails due to invalid course IDs, duplicate enrollments, or mismatched transaction details. It ensures the consistency and reliability of enrollment records by preventing logically impossible operations such as withdrawing from unregistered courses or enrolling twice in the same course. |

C. Package: com.wipro.cms.entity

| Class | Fields / Methods | Description |
|----------------|-------------------------------|---|
| Student | private String studentId | A unique alphanumeric identifier assigned to each student at the time of admission. It is used as a key reference in all academic transactions, including course enrollments and record retrieval. |
| | private String name | The official full name of the student. This information is used for academic reports, enrollment summaries, and administrative communication. |
| | private String program | Indicates the academic program the student is pursuing (e.g., BSc CS, BCom General). This field helps categorize students and provides context during course selection and academic tracking. |
| | private int currentCredits | Represents the total number of credits the student is currently enrolled in. This helps enforce institutional rules regarding maximum credit limits and prevents students from taking on excessive academic load. |
| | public getters & setters | These methods provide controlled access and modification rights to student attributes. Encapsulation ensures that student information remains accurate and tamper-proof across all system workflows. |

Class: Course

| Class | Fields / Methods | Description |
|--------|---------------------------|--|
| Course | private String courseId | A unique identifier assigned to every course offered by the institution. It distinguishes one course from another and ensures seamless reference during enrollment. |
| | private String title | The descriptive academic title of the course, such as "Data Structures" or "Financial Accounting". This name is displayed to students and administrators for easy course identification. |
| | private int credits | Specifies the credit weight of the course. Credit-based systems are widely used in colleges to measure academic load and determine tuition fees. |
| | private int maxCapacity | Determines the maximum number of students allowed to enroll in the course. This reflects realistic institutional limits based on classroom and faculty constraints. |
| | private int enrolledCount | Tracks the number of students who have already enrolled. It increases during enrollment and decreases when students withdraw. |
| | public getters & setters | These methods allow secure access to course information and ensure consistent data updates through well-defined channels. |

Class: Enrollment

| Class | Fields / Methods | Description |
|------------|-----------------------------|---|
| Enrollment | private String enrollmentId | A unique identifier generated for each successful enrollment transaction. This ID enables precise tracking and auditing of academic records. |
| | private String studentId | Indicates the student who has enrolled. This field links the enrollment to the appropriate student record for summarization and reporting. |
| | private String courseId | Specifies the course selected by the student. This allows lookup of course details such as credit weight and capacity. |
| | private String semester | Represents the academic term (e.g., SEM1, SEM2) during which the enrollment occurred. This helps organize and retrieve records chronologically. |
| | private double feeAmount | Stores the calculated tuition fee associated with the enrollment. The fee is based on the course's credit value multiplied by a standard rate. |

| Class | Fields / Methods | Description |
|-------|--------------------------|--|
| | public getters & setters | Provide safe access to enrollment attributes while maintaining the integrity of academic transaction data. |

D. Package: com.wipro.cms.service

| Class | Methods | Description |
|----------------|---|--|
| CollegeService | public CollegeService(ArrayList<Student>, ArrayList<Course>, ArrayList<Enrollment>) | Initializes the College Management System with dynamic lists of students, courses, and enrollments. Using ArrayLists allows the institution to expand or modify academic data without limitations. |
| | public boolean validateStudent(String studentId) throws InvalidStudentException | Ensures that the enrollment request corresponds to a valid student. Prevents enrollment attempts with unregistered or incorrect student IDs. |
| | public boolean validateCourse(String courseId) throws EnrollmentException | Ensures that the specified course exists in the catalog. Prevents invalid or outdated course selections. |
| | public boolean checkCourseCapacity(String courseId) throws CourseFullException | Verifies whether seats are available in the selected course. Blocks enrollment when capacity has been reached. |
| | public Enrollment enrollStudent(String studentId, String courseId, String semester) throws Exception | Executes the full enrollment workflow: validates student and course IDs, checks seat availability, prevents duplicate enrollment, calculates tuition fees based on credits, updates student credit load, increments course occupancy, and creates a new enrollment record. |
| | public boolean dropEnrollment(String enrollmentId) throws EnrollmentException | Allows students to withdraw from a course. Updates both student and course data accordingly. Prevents withdrawals from courses not actually enrolled. |
| | public double calculateFee(String courseId) | Computes course fees based on credits multiplied by a fixed institutional rate. |

| Class | Methods | Description |
|-------|---|--|
| | public void printStudentEnrollments(String studentId) | Displays a formatted list of all courses associated with the student, along with semester and fee details. |

E. Package: com.wipro.cms.main

```
package com.wipro.cms.main;
```

```
import java.util.ArrayList;
import com.wipro.cms.entity.Student;
import com.wipro.cms.entity.Course;
import com.wipro.cms.entity.Enrollment;
import com.wipro.cms.service.CollegeService;
import com.wipro.cms.util.*;
```

```
public class Main {
```

```
    public static void main(String[] args) {
```

```
        ArrayList<Student> students = new ArrayList<>();
        students.add(new Student("S001", "Rahul Sharma", "BSc CS", 0));
        students.add(new Student("S002", "Ananya Rao", "BCom General", 0));
```

```
        ArrayList<Course> courses = new ArrayList<>();
        courses.add(new Course("CO101", "Data Structures", 4, 2, 0));
        courses.add(new Course("CO102", "Operating Systems", 3, 1, 0));
```

```
        ArrayList<Enrollment> enrollments = new ArrayList<>();
```

```
        CollegeService service = new CollegeService(students, courses, enrollments);
```

```
        try {
```

```
service.validateStudent("S001");
service.validateCourse("CO101");
service.checkCourseCapacity("CO101");

Enrollment en = service.enrollStudent("S001", "CO101", "SEM1");

System.out.println("Enrollment Successful!");
System.out.println("Enrollment ID: " + en.getEnrollmentId());
System.out.println("Course ID : " + en.get courseId());
System.out.println("Fee Amount : Rs." + en.getFeeAmount());

System.out.println("\n--- Student Enrollment Summary ---");
service.printStudentEnrollments("S001");

} catch (InvalidStudentException ise) {
    System.out.println(ise.toString());
} catch (CourseFullException cfe) {
    System.out.println(cfe.toString());
} catch (EnrollmentException ee) {
    System.out.println(ee.toString());
} catch (Exception ex) {
    System.out.println("Unexpected Error: " + ex.toString());
}
}
```