HR Management System for SJSU

Composed By: The Architect

## I. Requirements

* The system should be able to let a student register with their SJSU ids.
* The system should be able to let the students view the courses that they can register for the upcoming semester.
* The system should be able to let the Lecturers view the students that are enrolled into their courses.

**II. Design Concept**

The HR Management system for SJSU includes OOP concepts such as inheritance, association, aggregation and composition. We have used Java programming platform (IntelliJ) to implement this system.

The HR Management system for SJSU has a parent class “Person” which includes attributes such as ‘first\_name’, “last\_name’ to indicate the name of the person and other details such as ‘phone’, ‘email’, ‘address’, ‘date\_of\_birth’, and ‘gender’. We are assuming that these attributes are necessary for the system, in order to get the person details.

The system has two classes, ‘Student’ and ‘Lecturer’ which extends the base class ‘Person’. The student class has few more specific attributes which defines a student identity separately from a lecturer. Student class has ‘sjsu\_id’ as a unique ID, ‘GPA’ to define his current academic score, ‘expected\_grad\_date’ will hold a date value of student’s expected graduation, and the ‘enrollments’ consists of a list of Courses the student has enrolled for. The students can enroll/register to the HR management system for a particular course via the ‘enroll()’ method and drop the same course using the ‘drop()’ method. The students can see a list of available courses at any time using the ‘listAvailableCourses()’ method by passing semester as a parameter.

The ‘lecturer’ class has ‘lecturer\_id’ attribute as a unique id, ‘salary’ with float as data type, ‘salary\_type’ which defines whether the salary will be given hourly, daily or course wise. The lecturer can view all the students registered to his classes by creating an object reference to the Course class, which in turn can call the ‘listStudent()’ method.

Furthermore, the ‘Enrollment’ class has the following key attributes, ‘enrollment\_id’, ‘sjsu\_id’, ‘course\_id’, ‘enrollement\_date’, ‘grade’ and a boolean value ‘is\_paid’ to define if a particular student has paid the fees or not. The student will be able to enroll to a particular ‘course\_id’ using the student's ‘sjsu\_id’.

The system has a class ‘Course’ with attributes such as ‘course\_id’, ‘course\_name’, ‘semester’, ‘fee’, ‘max\_student\_count’, ‘min\_student\_count’ (minimum and maximum number of students allowed to enroll to the course. ‘course\_enrollments’ is the list of all enrollment. The ‘lecturer\_id’ is the id of the lecturer who is teaching the course. The course class has methods such as ‘addStudent(Student)’ to checks if the student is eligible to enroll for the course based on the student's GPA, ‘dropStudent(Student)’ allows the ‘course’ class to drop a student and ‘listStudents()’ method which lists students enrolled for that course.

**III. Implementation**

Following are the technical aspects of the system including the implementation.

**a) Inheritance**

*Concept:* Ability to extend the class and inherit the functionality of base class.

Implementation*:* The HR Management system for SJSU has a parent class “Person” which includes the details of person. Moreover there are two other classes, ‘Student’ and ‘Lecturer’ which extends the base class ‘Person’ and *inherits* the properties of the class ‘Person’.

**b) Association**

*Concept:* Association is the *relationship* between classes. Classes have their own lifecycle and they have no owner, meaning that objects can be created and deleted individually.

*Implementation:* Student can enroll for multiple courses so there is a one-to-many association between student and enrollments. Course class can have multiple students enrolled, so there is a one-to-many association between the course and the enrollment class. One lecturer can also teach multiple courses so there is a one-to-many association between lecturer and course. Lecturer can also see the list of students that are enrolled into the course.

**c) Aggregation**

*Concept:* This is a specialized form of association but weak relationship between classes. A class has a reference to another class and simply possess has-a relationship. (e.g Car has an engine, Department has teachers)

*Implementation:* ‘Enrollment’ is a class which has a weak relationship with ‘Student’ class and ‘Course’ class. The ‘Student’ and ‘Course’ classes can still exist even if the ‘Enrollment’ class does not exist. Aggregation is also demonstrated between the lecturer and the course relationship. The ‘Lecturer’ class can still exist in our HR Management system even if there is no ‘Course’ class available.

**d) Composition**

*Concept:* This is a form of association having a strong relationship between classes. If a parent class is destroyed, all child classes also get deleted.

*Implementation:* Lecturer and Course classes possess composition relationship. The course cannot exist if there is no lecturer teaching the course. Besides, the Course also requires a minimum number of students to be enrolled. Enrollment and Course class also have a composition relationship, i.e, if there is no course available then the student cannot enroll. Again if the student class does not exist then the Enrollment class will not exist.