
# Model Driven Software Engineering

## COEN-6312

Instructor: Dr. Wahab Hamou-Lhadj

**University Management System**

**Deliverable 3**

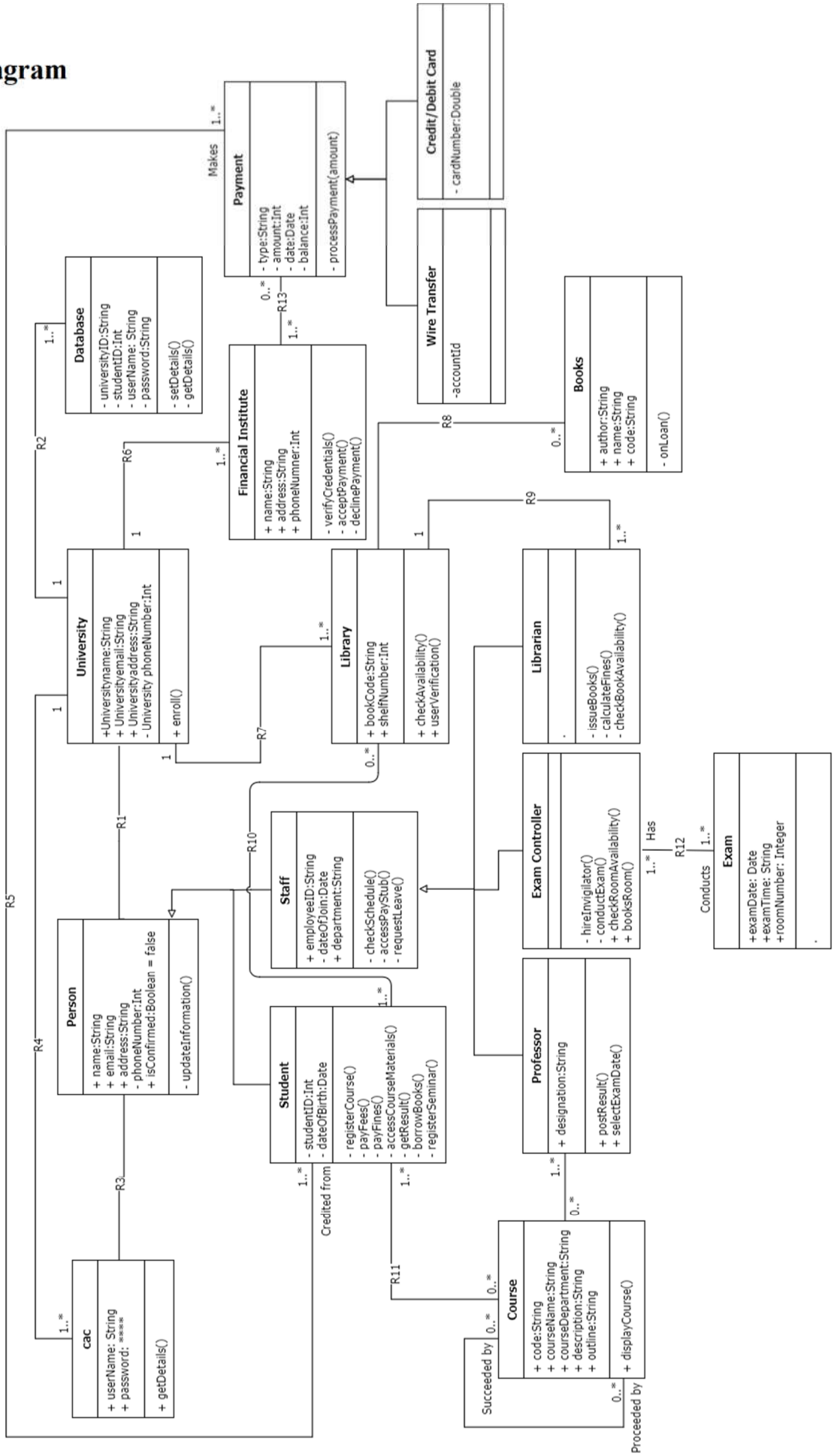
**Team: Hungry for Troubles**

Hem Raj Regmi 40015416
Bitta Rani Rana 40031066
Aeisha Vyas 40043598
Manjot Kaur Grewal 40014336
Harmanpreet Kaur Rajput 40014345

# Table of Content

Table of Contents

# 1. Class diagram



## 2. Key Classes, Attributes, and Association

In this design of a system, we have identified the key classes and group together as shown in the above class diagram which helped to determine the static relationship between them.

1. **Person class**: The class 'Person' is the generalized superclass of the subclasses 'Staff' and 'Student' which inherits identical functionality with the addition of the new characteristics of its own.

*Attribute*: The class is composed of the following attributes namely

| Attribute Name | Attribute Type |
|---|---|
| + name | String |
| + email | String |
| + address | String |
| - phoneNumber | Double |
| + isConfirmed | Boolean |

*Association:* The 'Person' class has a bi-directional association with the class 'University'. The person takes the role of 'member' in this association where many to one relationship exist.

2. **Database class**: The class 'database' refers to the organized collection of the information of the member of the university for easy access.

*Attribute*: The class is composed of the following attributes namely

| Attribute Name | Attribute Type |
|---|---|
| - universityID | String |
| - studentID | Integer |
| - userName | String |
| - password | String |

*Association:* The 'Database' class has a unidirectional association with the class 'University'. The University takes the role of 'has' in this association where one too many relationships exist.

3. **Financial Institute class:** The 'Financial Institute' class refers to the bank which is linked to the university when the student makes the payment for tuition.

*Attribute*: The class is composed of the following attributes namely

| Attribute Name | Attribute Type |
|---|---|
| + bankName | String |
| + bankAddress | String |
| + phoneNumber | Double |

*Association*: The 'Financial Institution' class has a bi-directional association with the class 'University' and 'Payment'. The 'Financial Institution' takes the role of 'linked' in this association where one too many relationships exist. And for the 'Payment' class many to many relationships exist.

4. L**ibrary class:** this class represents the library of the university that has a collection of books, which can be borrowed by the students.

*Attribute:* the attributes of the class are as follows

| Attribute Name | Attribute Type |
|---|---|
| + bookCode | String |
| + shelfNumber | Integer |

*Association:* The 'Library' class has a bi-directional association with the class 'Student' and a uni-directional association with 'Books'. The library takes the role of 'accessed by' in association with class 'Student' and multiplicity of zero to many, whereas it takes the role of 'has' in the association with class Books and has a multiplicity of zero to many.

5. **Books class**: This class represents the books in the university's library.

*Attribute:* the attributes of the class are as follows

| Attribute Name | Attribute Type |
|---|---|
| + author | String |
| + bookName | String |
| + code | String |

*Association:* The 'Books' class has an unidirectional association with the class 'Library'. The Library takes the role of 'has' in this association where zero to many relationships exist.

6. **Librarian class:** it consists of various details of the members working as Librarian in the university.

*Attribute:* the attributes of the class are covered by the intermediate class and the parent class, it inherits.

*Association:* This class is a child class of classes 'Staff' and 'Person'. The intermediate class is 'Staff' and the base class is 'Person'. It shares a bidirectional association with class 'Library' and takes the role of 'manages' with a multiplicity value of one only.

7. **Staff class:** This class represents one of the user types "staff" of the system. The staff has further three types of users into the hierarchy i.e. Exam Controller, Professor, and Librarian. These three types of users can use the system according to the constraints on the functions they can perform using the system

*Attribute:* the attributes of the class are as follows

| Attribute Name | Attribute Type |
|---|---|
| + employeeID | String |
| - dateOfJoin | Date |
| + department | String |
| - designation | String |

*Association:* The The 'staff' class has three subclasses where the staff is the superclass of its child class as 'Professor' class, Exam Controller class and 'Library' class.

8. **Student class:** This class represents the user types "student" of the system. The student uses the system to perform various tasks such as view course material, pay fees, etc.

*Attribute:* the attributes of the class are as follows

| Attribute Name | Attribute Type |
|---|---|
| - studentID | Integer |
| - dateOfBirth | Date |

*Association:* The 'student' class is a subclass of class 'person'. The 'Student' class has an association with class course i.e. the student can be enrolled in 0 to many courses and a course can be taken by 1 to many students. Also, the student class has an association with class Payment i.e. a student makes a payment and payment is credited to the student. The student in addition also has an association with class Library i.e. a student accesses a 0 to many libraries and a library is accessed by 1 to many students.

*9.* **Course class:** The course class represents the courses that are offered in a university which are taken by students or taught by professors

*Attribute:* the attributes of the class are as follows

| Attribute Name | Attribute Type |
|---|---|
| + code | String |
| + courseName | String |
| + courseDepartment | String |
| + description | String |
| + outline | String |

*Associations:* The courses class has a reflexive association with itself in that a course can be preceded by a course or a course can be succeeded by another course. The course also has an association with a class professor that the course is taught by a professor and a professor teaches a course. The Course class is also associated with student class i.e. the course is taken by students or the students are enrolled in a course.

## 3. OCL Expressions

Following are some OCL expressions corresponding to the key classes of the class diagram:

**1) This is the OCL command that checks database before a student is enrolled. Here the condition is, the student is enrolled only if he is not a student of the university.**

Context University :: enroll(S:Student):String

Pre: self.R2 -> exclude(S)

Post: self.R2 -> include(S)

**2) The user can login only with the valid username and password information.**

Context Person

Pre: forAll(self.R1.R2.username -> exist()) AND

Post: self.isConfirmed -> true.

**3) The statement verifying that the Person is the super class of class Student.**

Context Student

Inv: self.oclIsKindOf(Person) -> true

**4) The statement showing that a student is able to pay the fee only if the balance in his account is more than the amount to be payed.**

Context Payment :: processPayment(Integer Amount)

Pre: balance >= Amount

Post: balance@pre - Amount

**5) The expression showing that library keeps the count of the books borrowed by the student in given time. (*where onloan defines if the books borrowed by student from library)***

Context Books :: onLoan(B:Books)

Pre: onLoan -> notEmpty()

Post: Books = Books@pre + B

**6) Student can borrow maximum 5 books at a time from Library. (first check whether the books in the library)**

Self.R10 -> size() <= if (self.R10.bookCode -> exists()) then 5 endif

**7) Two exams should not have same room number.**

Context Exam

Inv: allInstances -> forAll (e1, e2: Exam | e1< >e2 implies e1.roomnumber < > e2.roomnumber )

**8) All the students must have unique student id.**

Context Student

Inv: allInstances -> forAll (s1, s2: Student | s1< >s2 implies s1.studentId < > s2.studentId )

**9)Each course can have at most 80 students and at least 10 students.**

Context Course

Inv: Self.R1 -> size () > = 10 AND Self.R1 -> size () < = 80

## 4. Contribution Table

| | Class Diagram | Key Classes, Attributes, and Association | OCL Expressions |
|---|---|---|---|
| Hem Raj Regmi | √ | Financial Institute, Report Preparation | √ |
| Bitta Rani Rana | √ | Person, Database | √ |
| Aeisha Vyas | √ | Library, Books, Librarian | √ |
| Manjot Kaur Grewal | √ | Staff | √ |
| Harmanpreet Kaur Rajput | √ | Student, Course | √ |

**GitHub:** https://github.com/sangamsyabil/COEN6312---College-Management-System