UNIVERSITÀ DI PISA

Department of Information Engineering

Information Systems
Task 0 Documentation

Students:
Adriano Botti
Antonio Le Caldare
Francesco Merola
Giacomo Ponziani

Contents

List of Figures

Application Specifications

The goal of the application that we implemented is to provide a way for both students and professors to manage the registration process for exams. More specifically, we want the application to exert the following functionalities:

- For Students:
    1. Check past exams results
    2. Register to an exam date
    3. Delete an exam registration
- For Professors:
    1. Add grades to an exam
    2. Create a new exam date

The application is realized using the Java language, with the JavaFX extension to manage a graphic interface. The back-end uses a MySQL database to store the information.

Requirements and Use Cases

Functional Requirements

The Professor:

1. shall be able to insert an exam, associated with a course he holds, in a date of choice
2. shall not be able to inster an exam for a date precedent to the current date
3. shall be able to insert the corresponding grade for a student in his registration for that exam.
4. shall insert all grades in the exact date of the exam

The Student:

1. shall be able to check the results of past exams
2. shall be able to register to an exam not yet took
3. shall not be able to register to an exam after the exam date.
4. shall be able to register to more successive exams for the same course
5. If the student registered to successive exams for the same course he just got a mark for, then those future registrations shall be deleted
6. shall be able to deregister from an exam he was previously registered to
7. shall not be able to deregister from an exam alredy took
8. shall not be able to deregister from an exam after the exam date.

Non-Functional Requirements

For the application we identified Consistency and Availability as the two most important non-functional requirements. Both the requirements can be satisfied by the use of a RDBMS, since for our application we don't manage high volumes of data. For this reason we employed a MySQL DBMS to implement our back-end

Use Case Diagram

From the funtional requirements, a Use Case Diagram is derived. For more detail and a step-by-step description of the different scenarios, see chapter *User's Manual*.

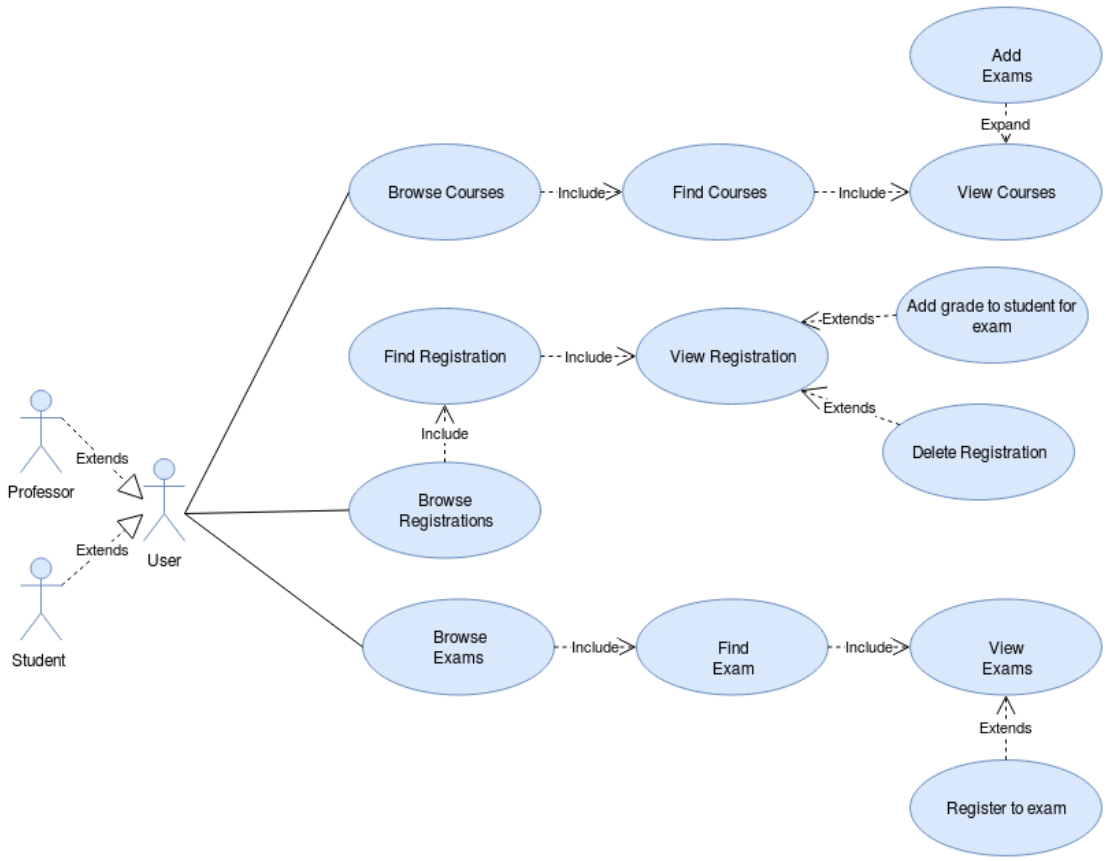

Figure 1: Use Cases Diagram

Entity-Relationship Diagram

ER Vocabulary

Names definition

Here we define in detail the terms for the main entites and relationships we will use in the following:

- *Student*
  A student is an entity which is able to perform the operations already defined in the requirements. He's an actor for our application.
- *Professor*
  A professor is an entity which is able to perform the operations already defined in the requirements. He's an actor for our application.
- *Course*
  A course is held by one and only one professor. The course object only includes information about its name, cfu and the helding professor. It holds no information about when exams for that course will take place.
- *Exam*
  An exam represents the actual date of the examination for a course.
- *Exam Result* An exam result relates an exam to all the students who registered to that exam, adding the information of the grade, if meaningful.

Entities

Entity	**Description**	**Attributes**
Student	Holds all the information related to the students	• $\underline{id}$ • $name$ • $surname$
Professor	Holds all the information related to the professors	• $\underline{id}$ • $name$ • $surname$
Course	Holds the information related to the courses	• $\underline{id}$ • $name$ • cfu • $professor$
Exam	Holds all the new and past exams	• $\underline{course(ext)}$ • $\underline{date}$

Relationships

Relationship	Description	Participants	Attributes
Teaching	Links Professors to their held courses	• Professor(1,N) • Course(1,1)	
Exam Result	Links students to exams, with the respective grade	• Student(0,N) • Exam(0,N)	• *grade*
Exam Date Creation	Links the courses to the exams through a date	• Course(0,N) • Exam(1,1)	

ER Diagram

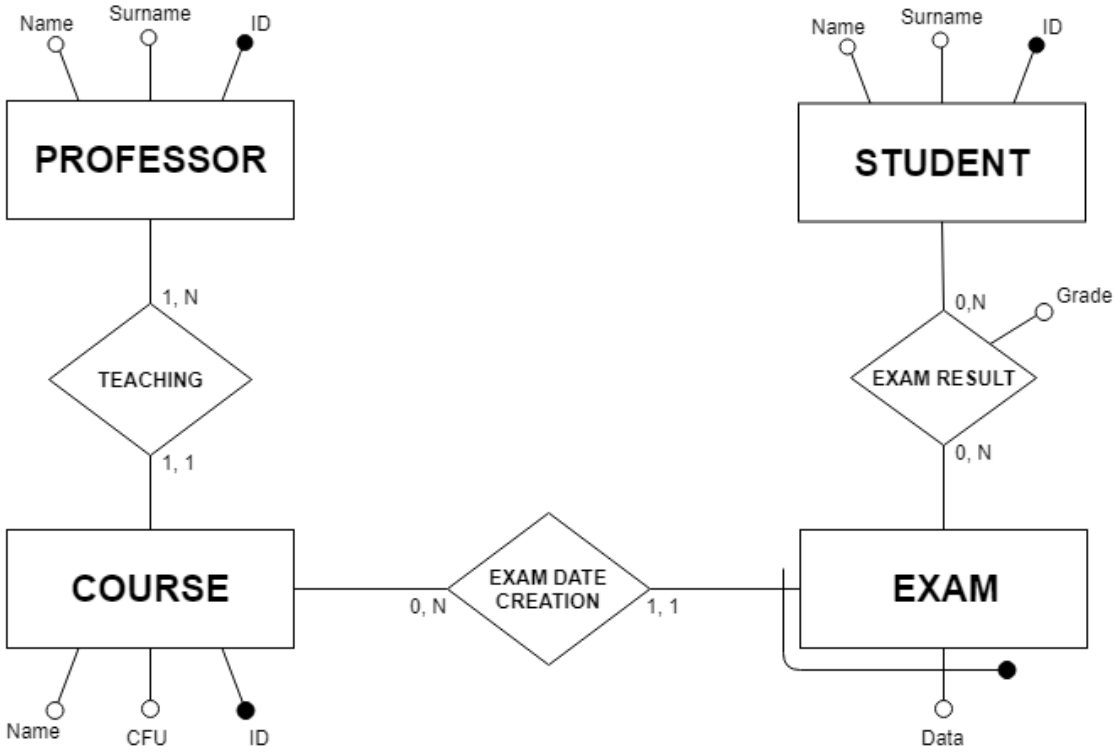

Figure 2: Entity - Relationship Diagram

UML Class Diagram

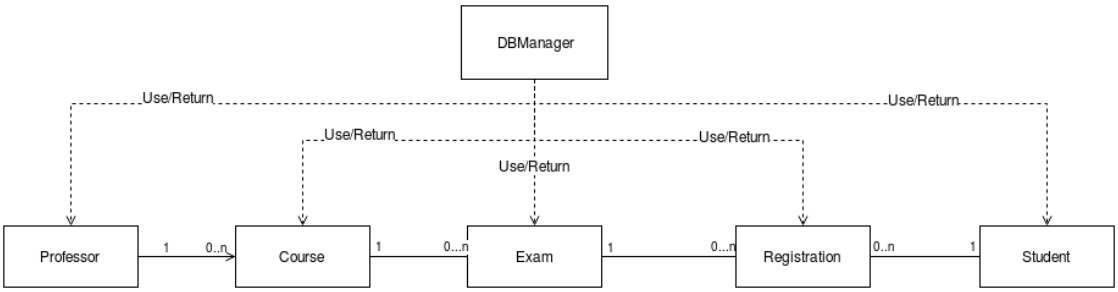

Figure 3: Entity - Relationship Diagram

User's Manual

Bibliography