# Course Management System DOCUMENTATION

## 1. INTRODUCTION

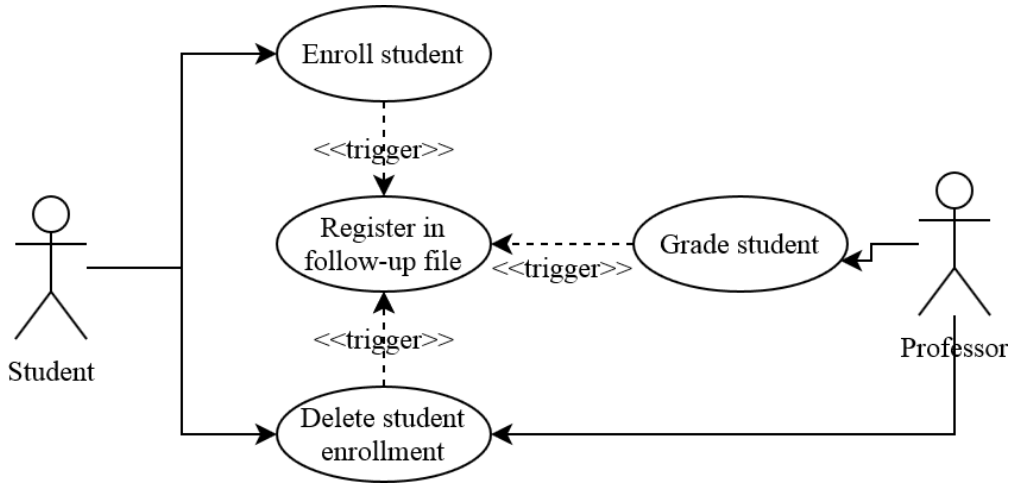
The requirements of the CMS are based on the work of [Baniassad, Clarke 04] and [Clarke, Baniassad 05]:

R1. Students can enroll in individual courses.
R2. Students can drop individual courses.
R3. Each student enrolling in an individual course must be registered.
R4. Each dropped course must be registered.
R5. Teachers can discharge students from an individual course.
R6. Each student discharge must be registered.
R7. When students are discharged from a course, they must be labeled as special.
R8. Teachers can grade student coursework.
R9. Each student grade must be registered.

We identified two types of actors–students and teachers–and three UCs: Enroll Student, Delete Student Enrollment, and Grade Student. Also, requirements R3, R4, R6, R7, and R9 were found to be cross-sectional with respect to the other requirements, since they all imply registering the resulting activity in a follow-up file. These requirements are encapsulated in an aspect identified as UC Register Follow-Up, and it is included in the rest of the UCs with stereotype <<trigger>>.

## 2. OVERVIEW DIAGRAMS

### 2.1. Use Case Diagram



### 2.2. Atomic ideas

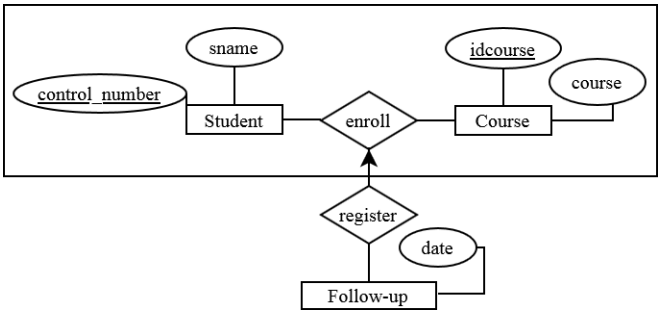
| Student |
| --- |
| control_number : Number<br>sname : String |

| Course |
| --- |
| course_number : Number<br>cname : String |

| Enrollments |
| --- |
| control_number : Number<br>course_number: Number<br>grade: Number<br>mark : String |

| Follow-up |
| --- |
| control_number : Number<br>course_number: Number<br>date: String<br>trans : String |

### 2.3. E-R diagram

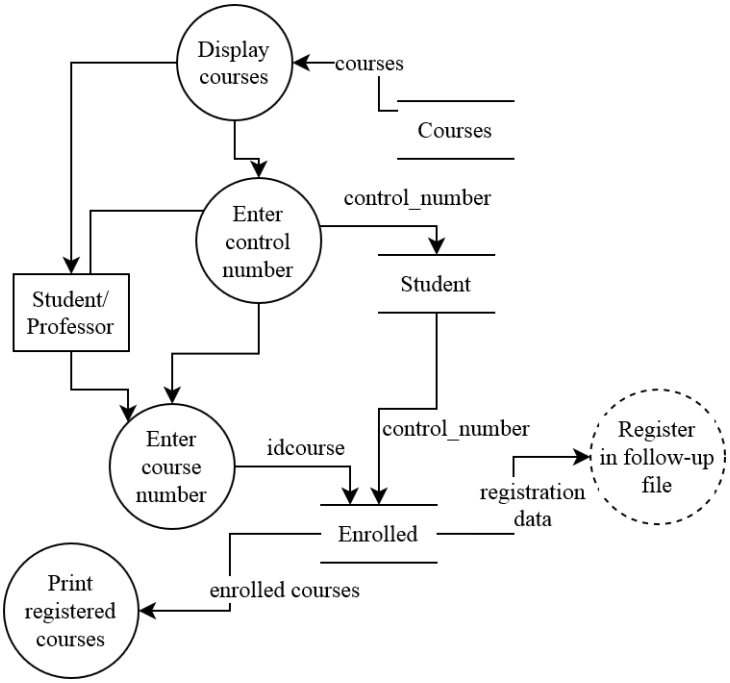



## 3. USE CASE ENROLL STUDENT

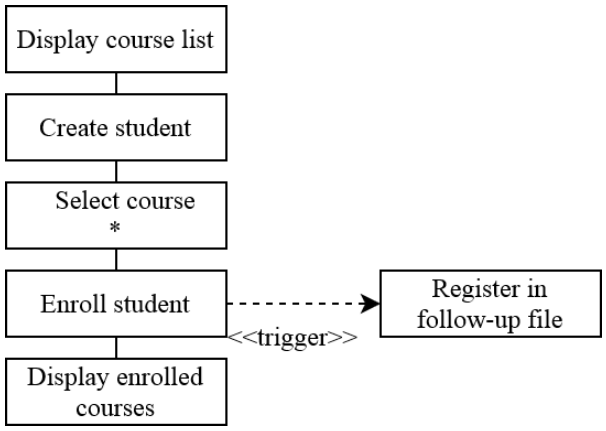
### 3.1. Description and prototype

| Course enrollment |
| --- |
| 3. NodeJS<br>2. JavaScript<br>1. MongoDB<br>0. Exit<br><br>Enter student ID number: 13011074<br>Select a course: 1<br>Successful registration!<br><br>Do you wish to enroll in another course? 1-Yes 0-No<br>0<br>[student ID number: 13011074 course:1] |

### 3.2. DFD




### 3.3. Structure diagram

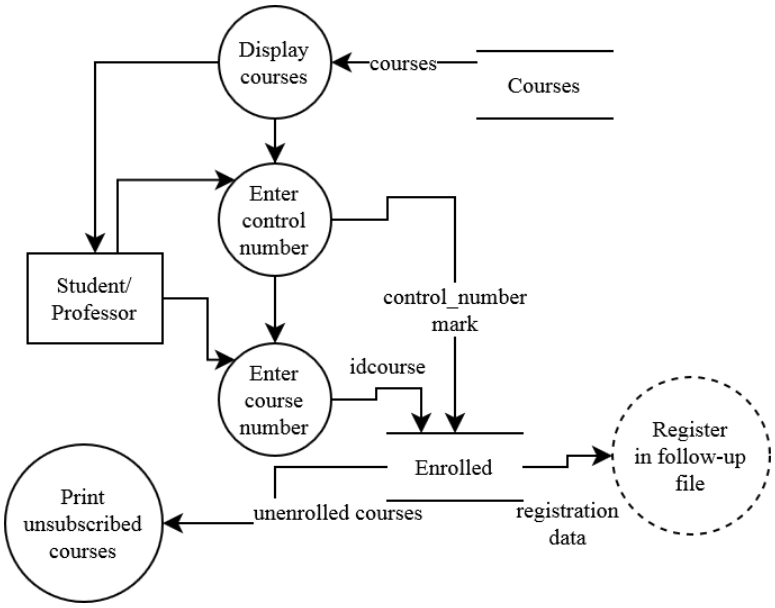



## 4. USE CASE DELETE STUDENT ENROLLMENT

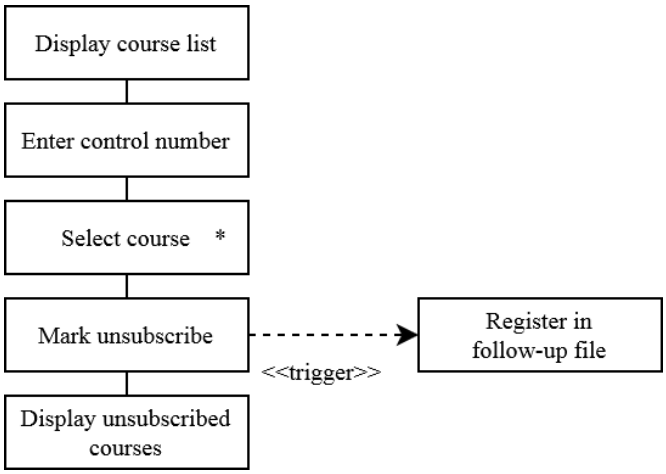
### 4.1. Description and prototype

| **Course unenrollment** |
| --- |
| 3. NodeJS<br>2. JavaScript<br>1. MongoDB<br>0. Exit<br><br>Enter student ID number: 13011074<br>Select a course: 1<br>Student has been unsuscribe!<br><br>Do you wish to unsuscribe another student? 1-Yes 0-No<br>0 |

### 4.2. DFD




### 4.3. Structure diagram




## 5. USE CASE GRADE STUDENT

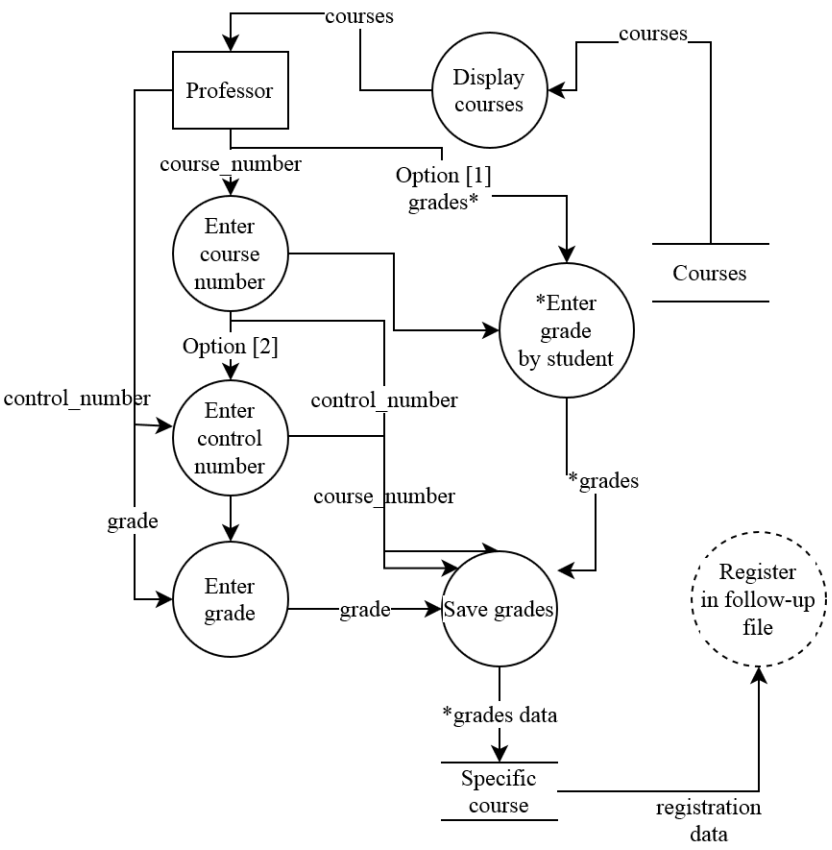
### 5.1. Description and prototype

| **Grading students** |
| --- |
| 3. NodeJS<br>2. JavaScript<br>1. MongoDB<br>0. Exit<br><br>Select a course: 1<br>Grade everyone [1], Select a student [2]: 1<br>Enter grade for 43257: 8.3<br>Enter grade for 89542: 9.4<br>Enter grade for 45629: 8.3<br>Enter grade for 34261: 5.6<br><br>Grades registered! |

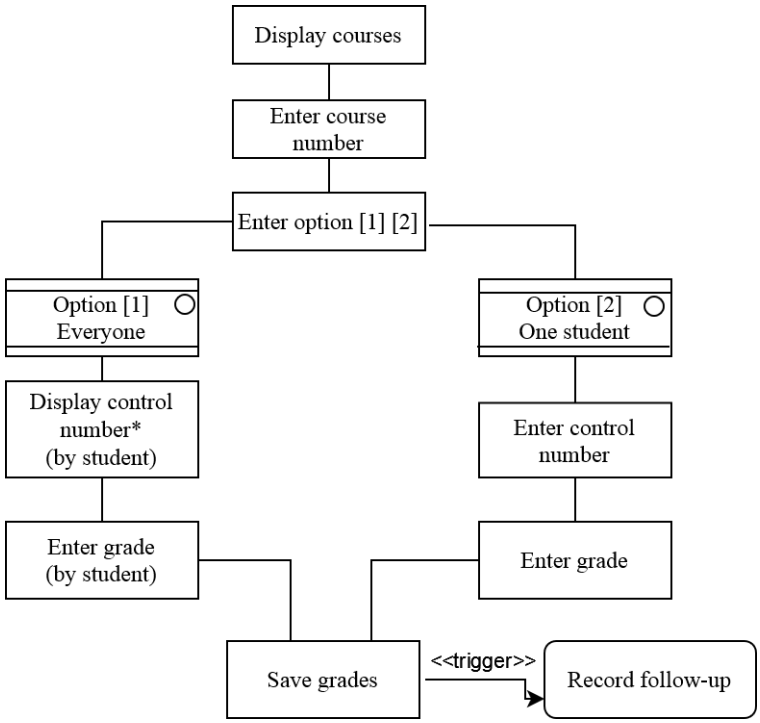
| **Grading students** |
| --- |
| 3. NodeJS<br>2. JavaScript<br>1. MongoDB<br>0. Exit<br><br>Select a course: 1<br>Grade everyone [1], Select a student [2]: 2<br>Enter the control number: 45629<br>Enter the grade: 8.3<br><br>Grade registered! |

### 5.2. DFD

Note: "*" means that the user must enter several data, in this case, several grades, one for each student.




### 5.3. Structure diagram

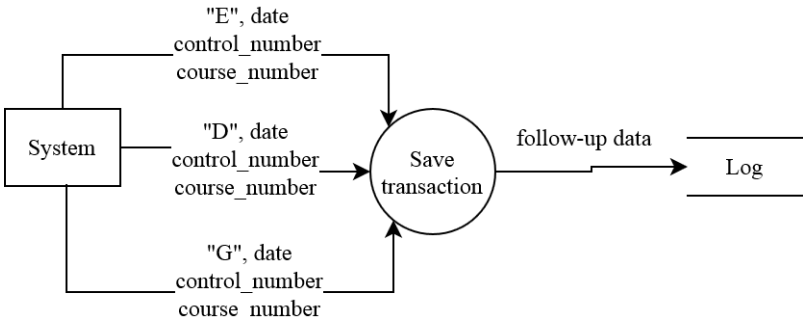


## 6. ASPECT REGISTER IN FOLLOW-UP FILE

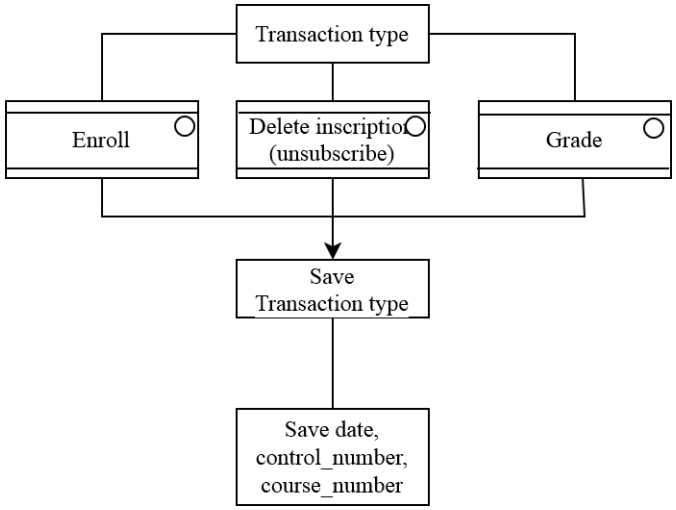
### 6.1. Description and prototype

There is no interactive interface for an aspect, the triggering is done through the Use Case who launch (<<trigger>>) the aspect, directly.

### 6.2. DFD




### 6.3. Structure diagram



## References

[Baniassad, Clarke 04] Baniassad, E., Clarke, S.: “Theme: An Approach for Aspect-Oriented Analysis and Design”; In Proceedings. 26th International Conference on Software Engineering, (2004), pages 158–167. doi: 10.1109/ICSE.2004.1317438. ISSN: 0270-5257.

[Clarke, Baniassad 05] Clarke, S., Baniassad, E.: “Aspect-Oriented Analysis and Design, The Theme Approach”; Object Technology. Addison-Wesley, first edition ISBN 0-321-24674-8. (2005)