Requirements Analysis Document

Lucía Asencio y Juan Riera

February 13, 2017

1. Introduction

   1.1. Purpose of the system
   The system is an application with educational purposes. It offers a communication and evaluation platform between professors and students.

   1.2. Scope of the system
   The application should have two types of users: students and teachers. There will be courses, created by teachers, which will be divided in units, subunits, subsubunits and so on. In each unit there will be exercises and notes all of which are organized by subjects.. Therefore it also serves as a database, that also provides a statistic calculation system detailed below.
   Students will have as many accounts as te number of students using the application, however, there will only be one teacher account for all teachers.

   1.3. Objectives and success criteria of the project
   The objective is to create a user-friendly application that meets the functionality detailed in the next section.

   1.4. Definitions, Acronyms and abbreviations
   Although sometimes in this document we will talk about units and sometimes about courses, we will be referring in both cases to the same thing.

2. System Description

   2.1. Functional Requirements The application has two types of users:

   2.1.1. User type 1: student

   2.1.1.1. Log in to the application

   2.1.1.2. View available courses and send an application to a course and get e-mail notifications when they are accepted or declined.

2.1.1.3. Access to each subject unit/subunit/subsub... and unit notes
Each item will only be visible after certain date (decided by teacher) if the student belongs to the course and is not expelled.

2.1.1.4. Access and solve exercises, each of which belongs to a unit, which belongs to a course. Exercises will only be visible and solvable after and before certain dates decided by teacher
A student may leave a question unanswered, and may quit the exercise without sending it whenever he wants, his answers will not be saved.
Once a student finishes an exercise, he cannot do it again.
Each exercise will have a relevance specified by the teacher, this relevance should be displayed at any time. After the deadline of an exercise, students can view their marks in that exercise (normalized from 0 to 10), as well as correct answers and the answers they wrote.

2.1.1.5. See their mark in a course up to that point, normalized from 0 to 10

2.1.1.6. Communicate with teachers and other students via e-mail

2.1.2. User type 2: teacher

2.1.2.1. Log in to the application

2.1.2.2. Get notifications via email when a student applies to a course.

2.1.2.3. Accept/decline students applications to a certain course

2.1.2.4. Set up courses, unit, subunits, subsubunits...
Also, write and add notes to each unit, subunit etc. This notes will be plain text.

2.1.2.5. Use an exercise maker to add exercises to units
This maker will set the unit where the exercise belongs, the two dates between which it is doable, the number of questions and the type of each of them (multiple choice/one choice/true-false/open answer), the order of the questions (which can be chosen to be random or a fixed order, also chosen by the teacher), the points of each question, the penalty for each wrong answer (if any), the value of the exercise for the final course mark (all of which don't ave to sum 10, since they will be normalized), the text of the question, the correct answer and the possible answers in the multiple choice/one

choice type of questions
Teacher can re-set up the whole exercise (excepting for the dates: only deadline can be changed and only if it is postponed), if no student has already done the exercise.

2.1.2.6. Make notes and exercises (and, optionally, courses) visible or invisible. This visibility may also be scheduled.

2.1.2.7. Access to all the information of a student, except for his log in data. This means teachers have access to every mark of every student, in every exercise and every course, every answer he has given to every question in every exercise he has finished. He also will have access to the course average marks of every student.

2.1.2.8. Access courses list.

2.1.2.9. View statistics
Statistics per subject : marks, average mark, failed vs. passed for each subject.
Statistics per exercise : marks, average mark for each exercise.
Statistics per question : not answered, answered, correct answers, wrong answers.

2.1.2.10. Expel students (and readmit them in the course), also access to a list of the expelled students

2.1.2.11. Communicate with students via e-mail

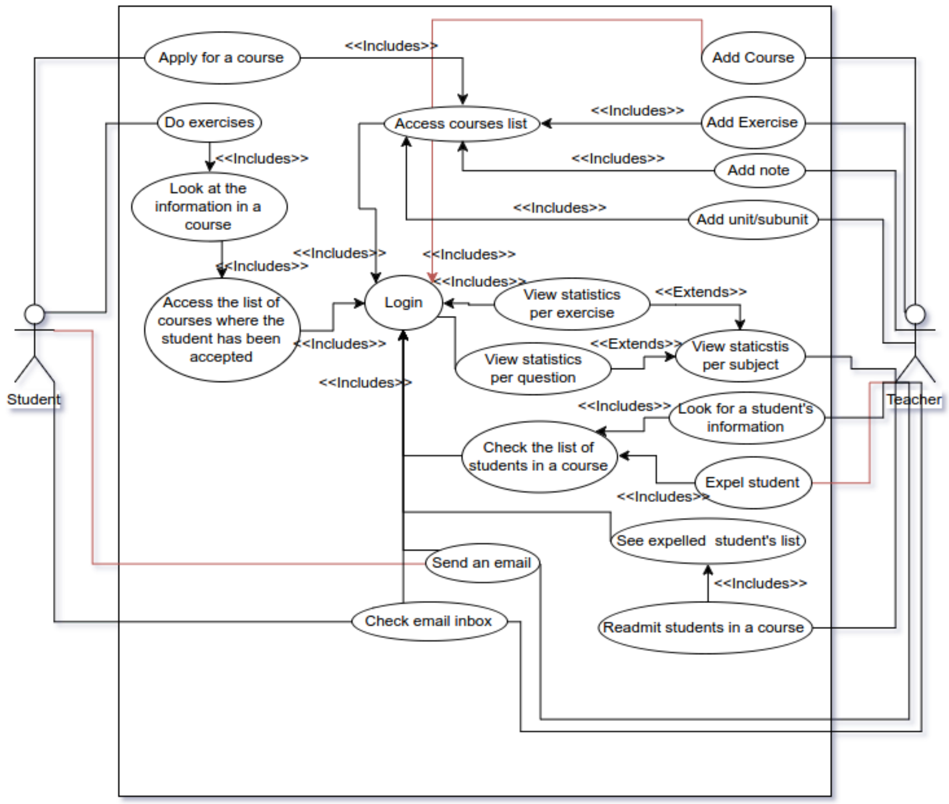
2.2. Non-functional Requirements

2.2.1. The application should be prepared to be used by the common user who does not have computer knowledge further than opening google.

2.2.2. The statistics calculations should not take longer than a few seconds.

3. Use Cases

3.1. Use Case diagram



3.2. Use case descriptions

3.2.1. Use case: Teacher

- Primary Actor
- Stakeholders and Goals
- Preconditions
- Success Guarantee
- Main Success Scenario
- Extensions
- Special Requirements
- Technology and Data Variations List
- Frequency
- Open Issues

3.2.2. Use case: Student

- Primary Actor
- Stakeholders and Goals
- Preconditions
- Success Guarantee
- Main Success Scenario
- Extensions
- Special Requirements
- Technology and Data Variations List
- Frequency
- Open Issues

4. Mockups