Internet Applications
Design and Implementation
2016/17
Project Assignment

João Costa Seco Jácome Cunha

Document versions

- October 12 – initial draft.

Contents

1

Problem Description

1.1 Introduction

The goal of the IADI project assignment in the 2016/17 edition is to design and implement a web application to manage college degrees, their courses, students enrolled, and all the information related (a clip-like application).

Consider the following data modelling aspects. Each degree is described by its basic informations, namely, designation, total number of ECTS, number of scholar years, and a set of courses.

Each course is identified by a course number (actually 2 letters and 4 digits), and further described by a name, a description, a number of ECTS, and a teaching team. Each course runs in different editions (fall or spring for each year) has a set of enrolled students. Note that only the name must not change between the different editions. Everything else can have different values in different editions.

A student is identified by a student number, and described in the system by its name, a personal email, the university email, address, photo, and degree and courses in which he/she is currently enrolled. Each student enrolled in a course may have a set of grades for the different evaluation parts (tests, assignment, exams, lab sessions, etc.).

Professors are identified by a username, a name, a department, and may be associated to a set of course editions (in different roles, professor, and teaching assistant).

1.2 Use cases

There are different actors in this application, namely:

Student students can visualize their information, update their address and photo. They can also add a secondary email and update it. They can also see information about courses and degrees.

Professor professors can update the courses that they manage. They can also enrol or dismiss students from courses.

Moreover, they can input evaluation to the courses and mark students for each part of the evaluation. They also give the final grade to students. They must also create new editions of courses. Courses are associated to a degree by this actor.

In the following we detail each actor and a few of its capabilities in the application.

1.2.1 Student

Students, before signing in, can:

- Register in the application and obtain a confirmation
- Login on the application and see their information

and after login in, can:

- Update photo
- Update address
- Add and update email address
- See all of its information (personal, concluded courses, current courses, grades, average, etc.)

1.2.2 Professor

Professors, before logging in, can:

- Login on the application and see their information

and after login, can:

- If they are assistant professors:

  - Mark students

- If they are professors:
  - Edit all of its details
  - Enrol and dismiss students
  - Create evaluation steps
  - Mark students

All users can logout after login in.

1.3 User interface

The user interface designed for this application should include pages to create, show, and edit all entities, a homepage where guests can see all the degrees available and their public informations, and a homepage that serves as dashboard for the professors where they can track each course (enrolled students, their marks, etc.).

1.4 Technical Requirements

The implementation of the suggested persistent data model and interactions completes the minimum requirements for this project. Moderate extensions in length shall be considered, both in data, as in functionality.

This project must be developed using HTML/CSS/JS, jQuery, React, and Java Spring. Variants should be first authorised by the lecturers.

1.5 Team and Interaction

This project should be realized by teams of exactly 3 (three) students. Teams with a different number of students must be validated by the lecturers.

The code must be delivered in git/bitbucket, where both lecturers are part of the project. The link to the repository must be send to the lecturers. In the readme of the repository you should include the team (name, number, and email id). Each phase delivery must be signalled by the students by including in the readme the git tag of the commit lecturers should look into. Note that the change to the readme will have a time stamp which should be earlier than the deadline established for the corresponding phase.

2

Phase 1

In this phase you must implement a responsive and reactive interface for your web application using HTML/CSS/JS, bootstrap, jQuery, and React. For now use static data in json/HTML files, as in classes. In Phase 2 we will use web services to replace that data.

2.1 Deliverables

For this phase you can reuse the project/code used during lab sessions and deliver it with the HTML/CSS/JS files implementing all the user interface defined before.

Moreover, you should also deliver the IFML specification of your application (only the student perspective) and the mapping between the IFML and the React components.

You must also deliver a small report describing the application delivered. The report must be written using LaTeX and it must also be in the repository (source and pdf files).

2.2 Important dates

Assignment due: October 28^{th} 2016 AoE (Hard deadline)

2.3 Evaluation

This delivery (code+IFML+mapping) is valued up-to 70 points, and the report is up-to 10, in a maximum of 200 points for the two planned phases.

3

Phase 2

In this phase you must implement the server side of the application using Java Spring.

3.1 Deliverables

You must deliver the server application as described, as well as the updated user interface if necessary.

You should also update the report, describing the data model and possible changes to the interface of the application.

3.2 Important dates

Assignment due: December 16^{th} 2016, AoE (Hard deadline)

3.3 Evaluation

This delivery is valued up-to 110 points, and the report is valued up-to 10 points, in a maximum of 200 points for the two planned phases.

4

Ethical issues

Your code must be original, and written by the members of the working group only. Any imported code, other than the one in the Java Spring framework, should be reported in advance to the lecturing team. Only well established libraries and frameworks will be accepted.