Project "CourseWeb"

Version 1.0

Preamble

The course projects are inspired by real-world needs, and they usually refer to similar sites already on the web. Students must follow the specifications given by this document, but they can also refine them through an interaction with the teacher and the analysis of similar websites. The final website must be completely original, well organized and easily accessible to all the users.

Site Specifications

After a decade spent trying to find the best way to organize online information about university courses, satisfying the wishes of students, teachers and administrative staff as well as the various regulations on this field, your teacher has (temporarily) run out of ideas, and decided to ask you to realize your " ideal teaching website", hoping to find new inspiration ☺

The *CourseWeb* site provides an online catalog of university courses, suitably simplified but still meeting the main requirements of transparency and completeness required for this type of website.

The site has to support **bilingual publication** (Italian and English) of all its content. Therefore you should have, for each view, at least two templates (one in Italian and one in English), and you should design the site so that it allows to enter information in both languages in the back-office and select which language to display in the front-office (*optionally*, if the content is not available in a language, the other language available will be always used).

Each course has to be necessarily associated to the following information. Any extra, arising from your personal experience, will be appreciated and perhaps may become an inspiration for the future.

- Basic data: name, code, scientific sector (SSD, for example INF / 01), language, semester
- Lecturers list
- Course description: prerequisites, learning outcomes, assessment method, teaching method, syllabus/analytical program (in general, one point for each credit)
- Textbooks (author, title, volume, year, publisher, web link if available)
- Relations with other courses: introductory courses, same-as courses, modules (for integrated courses)
- External links: course homepage, external resources, forum/eLearning
- Notes (for any detail that does not belong to one of the items above)

Optionally, the *learning outcomes* mentioned in the list can be structured according to the so-called *Dublin descriptors*, required by the European standardization process. The descriptors the course objectives to be described thorugh of the results achieved in five areas: *knowledge* (what knowledge is acquired through the course?), *Application* (how and where the knowledge arising from the course can be applied?), *Evaluation* (if and how the knowledge gained can help in evaluating, judging, comparing ...?), *Communication* (if and how students can share the knowledge gained?), *Lifelong learning skills* (the course provides tools to continue learning certain topics?).

Finally, the following information, while needed in a realistic course catalogue, in our project will be *optional*: the more you add, the more your project will have value.

- The list of degrees/curriculums where the course is taught, with indication of its value in credits (ECTS), and the respective type (A, B, C, D, F).
- List of teaching support items (with name, description and size) that can be downloaded from the website (such as PDF documents).

In order to create a "virtual guide" **the system must collect all this information year by year**. This means that you should **create a system where all the information above is associated with an academic year**, and there can be several copies for different academic years.

The following list contains a brief description of the contents and functionalities required by this site. Obviously, any further refinement or enrichment of these specifications will increase the value of the project.

- The system has three types of users: *anonymous*, *teacher* and *administrator*. Only administrators can register new users and assign them a type.
- All the users can view the complete list of courses, possibly filtered by name (also partial), code, SSD, semester, teacher, language and degrees for which it is available (if you have modeled such information). Clicking on a course will display its full record.
- Of course, the course page must be very well-finished, to make the contained information as clear and accessible as possible. You can split the data across multiple pages, remembering, however, that the essential information must be accessible with a small number of clicks.
- Site administrators can create courses (with at least the essential information: name code) and assign them to the teachers.
- Teachers can enter and edit all information belonging to their assigned courses, while administrators can modify all the courses.
- The back-office for teachers should also be very intuitive. It may be useful to provide contextual help to clarify the meaning of the various fields, and possibly split the course information into logical segments to be compiled in a wizard-style process. Where possible (e.g., in the selection of related courses, of languages, semesters, SSD, types of credits, etc.) the input should be assisted or guided.
- The site must have a log where all transactions carried out through the back-office are recorded (such as "User X has changed the course Y": you can decide which level of detail to include in these entries).
- Finally, to realize the "virtual guide" described above, the site should behave as follows:
  1. the information presented in a course page are, by default, those related to the last academic year (which must be indicated on the page) where the course has been updated.
  2. the user should be able to choose an academic year and read the course information related to such academic year, if available (e.g., through controls on the course page, or using suitable filters on the course list).
  3. in the back office, the information entered or edited by the teacher are always related to the current academic year. When a teacher tries to change the information of a course, and no information is available for the current year, the information belonging to the closest past year are automatically copied to the current year. *Optionally*, administrators may be able change the course information for any academic year.

Technologies

- The basic structure of the site must be created using HTML5 (*with XML or Polyglot syntax*).The validation of all the site pages with respect to the chosen HTML flavour is an important part of the development and **must** be reported in the documentation.
- The site layout must be realised using CCS style sheets. The layout can be freely based on third party layouts available on the web or shown during the lectures. In this case, the degree of **customization** of the layout will be taken into account in the final project assessment. **A responsive layout is not required, but strongly suggested.**
- For the *client-side* programming, JavaScript is the **required** language. It is possible to include libraries developed by third parties, provided that they have suitable cross-browser portability and that they are described in the project documentation. However, **any abuse of these technologies is not recommended**, especially when they can be replaced with a suitable use of HTML, CSS, etc. **In general, your site should be functional also with JavaScript disabled**. The site without scripts may be less "friendly" or allow the access to "core" functionalities only, but sites whose dynamics is entirely script-based are not allowed. However, scripts can play a more important a role in the functionalities whose users are restricted and pre-determined (e.g., in the *back-end* functionalities for administrators, but not in the public *front-end* or in the login procedure).
- For the *server-side* programming, Java *(servlets, JSP)* is the **required** language. Any DBMS and template engine can be employed, if required. Again, it is possible to rely on external libraries.
- In general, the site must work and have a good *rendering* on Internet Explorer/Edge, Mozilla Firefox and Google Chrome, and *possibly* be compatible with older browsers (in this case it should at least *degrade well)*and with the latest versions of other browsers, like Opera. Browser compatibility **must** be explicitly stated in the documentation.

Project Development and Documentation

The specifications may not be exhaustive or completely defined. Every feature added or refined, also through an interaction with the stakeholder or the site end users, will be adequately assessed. All the design choices must be discussed and motivated.

The final project, developed following the guidelines given by the present specification, must be a fully functional website, whose contents and features will be assessed during the examination. The specification parts marked as *optional*, if not developed, will not make the project insufficient but, on the other hand, will not allow your project to reach the highest mark. If you choose to implement an optional feature, the result should not be necessarily perfect or complete: it must only show your commitment to deal with an advanced issue.

The documentation (**in electronic format**) which accompanies the project **must** contain at least the following information:

- Indication of software dependencies (which libraries are required on the client and server side?).
- Indication of the functionalities that have been developed or not developed. Detailed description of any extra or optional functionality added to the project.
- Diagram showing the site structure.
- Relational schema of the database (if any).
- Analytical description of the site layout, also indicating its static/dynamic components.
- Description of any advanced technology (language, framework, plugin, library, etc.) used in the project, specifying the reason of its adoption and the actual contribution given to the project development.

- Description of *any* cross-browser programming/rendering problem encountered, list of compatible browsers.
- Screenshots of the most important website pages (*optional*).

The actual contribution of each group member to the project **must** be declared in the documentation (indicating, for example, the members that mainly worked on server and client side programming, the layout designer, etc.). During the examination, each group member will describe its part of the realisation.

Project Evaluation

To evaluate the project, the following issues will be considered (in order of importance):

1. Compliance with the specifications.
2. Technical correctness.
3. Organizational clarity and correctness of the contents.
4. Accessibility and standards compliance.
5. Appropriate use of static and dynamic contents.
6. Design quality.
7. Adequacy of the documentation.

This evaluation will be combined with the result of the project discussion.

Additional Information

This specification is available in PDF format on the website of the Web Engineering course at the address http://www.di.univaq.it/gdellape.Additional information on the specifications can be obtained directly via email by writing to giuseppe.dellapenna@univaq.it.

Please note that projects should be carried out by *small* student groups (three members is the recommended number). Exceptions to this rule must be agreed with the teacher.

Summary

This summary should be compiled and sent to the teacher, in electronic format, *before* the examination. A copy of this form should be also attached to the project documentation.

Project name: ______________________________

Authors:

First Name	Last Name	ID	Contribution / Role in project development

Client-side technologies/libraries/frameworks:
(HTML5, CSS 2/3, Javascript, JQuery, Bootstrap…)

____

____

Server-side technologies/libraries/frameworks:
(e.g., JSP, MySQL, JPA, template engines…)

____

____

Compatible browsers:

Browser	Version	Compatible	Degrading	Not compatible	Not tried
Internet Explorer					
Edge					
Mozilla Firefox					
Google Chrome					
Opera					
Safari					

Project discussion date: ______________________________