Projects

USAL1J 2014-2015

Projects will be done in groups of 2 or 3 (maximum)

Projects presentations will take place in the week of 1/4/2015.

Your projects will be hosted on `gforge.cnam.fr` (I will create the accounts for you, as well as the svn repositories).

The presentations will contain two parts : a general demo of your software, and individual questions, asking how some functionalities are made. I may, ask you, individually, to add a simple functionality to the software (you will have one hour to do so).

Project 1 : Tickets management system

You are asked to develop a system for assigning and solving users related problems for a software company.

Users of the application will be employees of the company.

When a problem is detected (either by someone of the company or by a client), it will be registered.

Any employee with an account can register a problem.

The problem will have at least the following informations:

- title
- description
- product involve (a software in the company)
- importance level
- date of creation
- a state in its processing

When a problem has been registered, employees with «management» rights will be able to assign them to a particular user.

Each user will be able:

- to see what problems he is supposed to process
- to see what problem exist for the products he «follows»

When he connects, he will be able to see what are the recent actions on the products he follows

When a user works on a problem, he will be able to

- add information to it (files or text) to document his work
- change the problem status:
  - to be done
  - in work
  - testing
  - solved
  - transfer request : the user ask the original manager to transfer the problem to another user. [this is an advanced functionality. Wait until

you have done most of the rest to work on it]
    - rejected (it wasn’t a problem after all)
- The manager who assigned the ticket to the user is the only one who can «close» the ticket.
- The complete history of each problem must be kept
- There should be a search system in order to get tickets
    - by words in description
    - by products names
    - by users
    - by dates

Project 2 : Cooking web site

The goal of this project is to create a web site devoted to cooking, where users

- can enter their own recipes
- the recipes must have a reasonable representation of the needed ingredients (see next question)
- A user can enter the content of his fridge and can get a list of recipes compatible with the fridge content
- *advanced functionality*: sort recipes according to their compatibility with the fridge's content (minimizing the ingredients to buy).
- it's possible to browse the recipes
- it's possible to search recipes with various criteria (in particular ingredients, but you can add others, like country, type of cooking (grill, cold, oven...)

Project 3 : “Intelligent” classroom timetable system

The idea is to create a model of a time table application for a high school, a bit like http://emploi-du-temps.cnam.fr.

Connected users can modify the time table. Visitors can only see them.

The constraints are the following:

- the display of the time table as an actual table is wished, but is less important than the management of the rest of the software. If you are a group of 3, one of you may work on this issue. But don’t block on it. I’d rather have a working simple interface than a nice display with no working code behind it.
- The system must be able to deal with:
  - Groups of students (you don’t deal with individual students here)
  - Courses
  - Classrooms
  - Teachers
- A course session will be a given course, given at a given time span on a given day to a certain group of students, in a classroom, by a certain teacher. For practical purposes, some of this information might be optional (for instance, we might not know the name of the teacher yet).
- Groups of students and classrooms have size. A group of 40 students can’t take a class in a classroom of 20 seats.
- The system should be able to warn the user when he is creating a course with logical impossibilities (for instance, the classroom is already occupied).
- Registered users can delete or modify existing course session, and, for instance, move them to different days.
- Advanced functionality: for courses with logical impossibilities, automatically propose a change that the user can accept to fix the course. Don’t work on it until the rest of the code is ok.

Project 4 : Programming project : MCQ

In this project, you are supposed to implement a multiple-choice questionnaire system using Java and JSF. The web site will be visited both by professors and by students.

- professors can create questions, and build questionnaires regrouping a number of questions. They want to be able to automatically evaluate the students answers.
- Students can take the MCQs, answer them, and get their marks.

Rules

- Each user must be authenticated, and has a status (student, professor), and an email.
- A question has a unique id, a theme, an author, a question text, and a number of proposed answers (and a way to tell which answer is correct).
- A MCQ is a list of questions, with a title.

What the site should do

- all users should be able to log in.
- professors
  - can post questions and answers
  - can add a theme (for instance “graph algorithms”)
  - can find questions for a particular theme
  - can select questions to add them to a new MCQ
  - can choose to “publish” a MCQ (an unpublished MCQ won't be seen by students). A published MCQ has a deadline (students won't be able to post answers after this deadline).
  - can see the marks of the students
  - can publish the marks of the students to a MCQ.
- Students
  - can read/fill/send a MCQ.
  - can view their personnal results to MCQs.

In this version, we are not interested by the details of the students answers, and we won't keep them. Only the mark will be kept.