CoursePicker Team Project

# CSCI 4300

Introduction: The UGA Registrar’s Office publishes a series of “static reports” on the status of all classes offered at UGA. These reports are published daily in computer-readable form, and provide a “snapshot” of all courses available for registration. You can see a list of relevant static reports at https://apps.reg.uga.edu/reporting/staticReports.

In this project, you will build a Course Picker application to allow UGA students to look for courses to satisfy specific course requirements. Your team will be doing the data model, Java specifications, and user interface design, so I won’t cramp your creativity by showing examples. However, here are the key features I would like to see in your application:

* Users should be able to pick from a list of UGA requirements and get a list of all available courses that meet the selected requirements;
* Users should have a graphical representation of their weekly schedule, created on the client sice using the HTML 5 “Canvas” element and drawn using JavaScript. This should include the ability to add and remove courses from the schedule;
* The application should have the ability to detect when the user selects a course that has a time conflict with a course that is already on the schedule, and resolve the conflict either by not adding the new course or by removing the old course.

# Iterations of this project:

* The first iteration will be publicly available. Users will create a session-scoped class schedule that evaporates when they leave the app;
* A possible second iteration would maintain user profiles, protected by a UGA MyID login.

The second iteration is a challenge exercise! Your team can get full credit by completing the first iteration.

# Teamwork setup and grading

* In order to discourage free riders, each student’s grade will be based on a combination of the overall project quality (50%) and the student’s individual contribution (50%). Therefore, we need to clearly document each student’s contributions. The unit of contributed work can be as small as a Java or Javascript function, but it must be an individual contribution. The HTML content of a JSP page may be defined as a work unit independent of its JavaScript content.
* The author of each unit should be clearly identified in the spec comment for that unit (shell script, class or function). Unfortunately, the Javadoc “@author” tag cannot be used in a method spec, only in a class comment.
* Group design work is encouraged. The entire group should meet and discuss the user interface, the database model, and the specs for the various Java and JavaScript functions you will be using. Only the JUnit test code and the Java and JavaScript implementation code needs individual authorship.
* Create a [Work Breakdown Structure](http://en.wikipedia.org/wiki/Work_breakdown_structure) to allocate the various tasks among team members. An Excel worksheet is sufficient for the WBS of a small project like this one. Each work package should be assigned either to a team member or the entire group.
* Once coding begins, use the **git** version control system to manage your code. We’ll be learning this system in class.
* There is one part of this project that will be a collaboration among all the teams: listing the courses that will satisfy each requirement. I will divide up the requirements among the teams; each team will come up with a CSV-formatted file with the format

“Requirement”,”Course”

--suitable for uploading into a MySQL database table using the LOAD DATA INFILE command. In this way, each team can build its own database table containing all the courses that might be used to satisfy a given requirement. We are deliberately ignoring complexities such as courses that can only be used under certain circumstances, requirements that need more than one course to satisfy them, *etc*.

# Milestones and project deadlines

* I will ask for preliminary submissions along the way, and will make a maximum attempt to respond with helpful comments within two days’ time. Except for the final submission deadline, your team will be able to resubmit a component within two days of receiving my comments.

|  |  |
| --- | --- |
| **Milestone** | **Submission Deadline** |
| Requirement-course CSV file | Mar 25 |
| User interface mockup (HTML files, no JavaScript) | Mar 25 |
| Work Breakdown Structure | Mar 25 |
| Data model | Mar 25 |
| Java and JavaScript specs | Apr 1 |
| Final code submission | Apr 26 |
| Student demos | Apr 29 |

Please don’t ask to continue working on this project past Apr 29, the last day of classes. You need to be studying for final exams, and I need to be grading your work!

# Grading Rubric (Team Grade)

|  |  |
| --- | --- |
| **Capability** | **Points** |
| Daily refresh of courses | 10 |
| Pick courses satisfying a requirement | 10 |
| User-friendly display of course list | 10 |
| JavaScript graphical display | 10 |
| Handling time conflicts | 10 |

# As you might expect, all code will be subject to grading deductions for failure to follow the course coding guidelines (spec comments, loop comments, informative variable names, *etc*.)

# Grading Rubric (individual work)

This will be based on the fraction of the work done by each individual. My rough metric will be lines of code, but I am willing to subjectively tweak this measure up if your contribution was particularly challenging. If you contributed at least 20% of the total work, you will get full credit.

If possible, each student should create some JUnit test code, some Java implementation code, and a JavaScript function. The test code and implementation code for a class should be written by different people.

# Academic Honesty Guidelines:

Since this is group assignment, you are allowed to help others on your team with code. However, please don’t write code under another student’s name.

# Final Exam question related to this assignment:

On the final exam, you will be asked the following question:

*“Discuss a design decision that your team made during the CoursePicker project. Explain the tradeoffs involved in this decision and why your team made the decision it did.”*

This question will be an entire section of the final exam, accounting for about 10% to 20% of the exam grade. As you can see, it is a team question, and you are encouraged to discuss it with the rest of your team. I won’t consider it cheating if I get similar answers from all of the team members!