Project 1 Part C

Due Friday, May 4th, 2012 by 11:00pm

The final part of the Movie Database project is very open-ended and some students may want to get started early.

In the last part you are to (finally) create a fully functioning Movie Database system accessed by users exclusively through a Web interface. The functionality of your Movie Database system is quite flexible, although we expect all students to implement some baseline capabilities:

Three input pages:

- Page I1: A page that lets users to add actor and/or director information. e.g. Here are some "reasonable" names: Chu-Cheng Hsieh, J'son Lee, etc.
- Page I2: A page that lets users to add comments to movies.
- Page I3: A page that lets users to add movie information.
- Page I4: A page that lets users to add "actor/director to movie" relation(s).

Note that when your PHP scripts get an input from the user, your program will likely need to lookup either MaxPersonID or MaxMovieID table, assign a unique ID to the new actor/movie, and increase the MaxPersonID or MaxMovieID table value.

Two browsing pages:

- Page B1: A page that shows actor information.
  - Show links to the movies that the actor was in.
- Page B2: A page that shows movie information.
  - Show links to the actors/actresses that were in this movie.
  - Show the average score of the movie based on user feedbacks.
  - Show all user comments.
  - Contain "Add Comment" button which links to Page I2 where users can add comments.

One search page:

- Page S1: A page that lets users search for an actor/actress/movie through a keyword search interface. *Your search page should support multi-words search. For example, search for "Tom Hanks". (You can interpret space as "AND" or "OR" relation) Although the order in which the elements of a SET are listed is irrelevant, unlike a sequence or tuple, you may sort them before present to users for their convenience.

Reminders

1. It is okay to embed some javascript or CSS to make your website astonishing, but please make sure your work is **completely self-contained**. Namely, all javascript/CSS files related to your website should be incorporated in your submission. However, we offer only applause but nothing else nor extra points to your visual design.
2. "Pages" mentioned above are conceptual blocks to users, not "files". For example, it is okay to use tabs

to switch between pages, and put all pages in one file.
3. If your work requires more than 5MB space, you have to get approval from TA(s) in advance.
4. You may assume that the user is not a hacker, and thus there will be no SQL injection, etc.

Hints on adding a clickable link to your page:

Note that your php pages can be accessed directly through a URL like http://guestip/~cs143/mypage.php?param1=value1&param2=value2&param3=value3 where mypage.php is the name of your php page, and the string after the question mark contains parameters passed to the php page just like in a "GET" method. So you can embed a clickable link to the page generated by your php code using hyperlinks of the above format, with appropriate parameters that instruct your PHP code to execute the right query.

Notes:

- Any PHP/database errors due to data entry errors or bad input values should be managed gracefully, i.e., it should be possible for users to continue interacting with the system, and the database should not be corrupt.
- You need to use the same set of tables (and only them) that you created in Part B of this project. You should use the database CS143 by connecting with username "cs143" and empty password.
- Don't forget to put some effort in the design of your Web interface. Though specifics on the ease of use of your web site are up to you, the user should be able to navigate through your web site without difficulty.
- Some of the pages described above may be combined. For example, a search box may appear in browsing pages B1 and B2, in effect combining page S1 with browsing pages B1, B2.
- A demo site is available at http://oak.cs.ucla.edu/cs143/project/demo/p1c/movie.html. This page is available strictly to give you an idea of the basic requirements, and is *not* meant to guide your choice of style or user interface in any way. Please be creative, and do not simply mimic the UI of the demo site.

In adding user reviews to a movie, you may need to obtain the current timestamp. You can do it either in your PHP code or in MySQL itself. The MySQL date and time functions page explains how you can obtain the current timestamp in MySQL.

While the functionality of your Movie Database system is quite open-ended, the interface itself is *extremely* open-ended. This class is not a user interface class and you can certainly get full credit for a solid system with simple input boxes, menus, and/or radio buttons, and simple HTML output tables. However, we welcome much snazzier interfaces, like something resembling IMDb.com itself, or perhaps even something better!

Although the project interface is very open-ended, we have to be certain the features of your system are accessible in a typical browser environment. For this reason, you must make sure your system work in Mozilla Firefox 2.0 or greater without any additional plugins or extensions. If you feel this restriction will somehow limit a feature you with to implement, you must get "preapproval" from the TA to use a different browser environment. Send an email message to your TA telling him precisely what browser environment you wish to use for your project and why it is necessary.

When the preapproval process is not followed, projects that have problems on Mozilla Firefox may lose points, possibly all points if we cannot run your project at all.

We may select a small number of Movie Database systems to be demonstrated to the class at time to be determined. Students will not receive extra credit, but they will receive extra recognition. The criteria for selection will be some combination of beyond-the-basics functionality and a good user interface.

Late Submission Policy

To accommodate the emergencies that students may encounter, each team has 4-day grace period for late submission. The grace period can be used for any part of the project in the unit of one day. For example, a student may use 1-day grace period for Project 1A and 2-day grace period for Project 2B. Any single project part may not be more than 2 days late. Note that even if a team submits a project 12 hours late, they would need to use a full day grace period to avoid late penalty. If your project is submitted late, we will automatically use the available days in your grace period unless you specifically mention otherwise in the README file.

What to Submit

- A README file, with a detailed explanation of which of the project criteria you met, as well as any additional features you have implemented.
  - If you worked as a team with a partner, briefly explain how you split the work and collaborated in your README file. Also briefly explain what aspect you feel you can improve in a team setting for better collaboration.
- All the source code and other supplementary files in a single zip file, so that it can be deployed and run by the TA on the VMWare image.
  - Remember to use relative URLs in your machine/port references. You may be get zero point for code with faulty URL links that don't work during grading.

Visit the Project 1C submission page to submit electronically by the deadline. In order to accomodate the last minite snafu during submission, you will have 30-minute window after the deadline to finish your submission process. That is, as long as you start your submission before the deadline and complete within 30 minutes after the deadline, we won't deduct your grade period without any penalty.

Grading Session

Your project 1C submissions will be graded through "grading sessions," in which each student group should demonstrate their implementation of the project to the TAs in person. Every student group will be allocated 7-minute time slot for demonstration. A sign-up sheet for the grading sessions is available online.

Here are important things that you will have to remember before coming to your grading session:

1. During your grading session, the TA will simply unzip your submitted file into a directory and ask you to demonstrate the required functionality. (The CS143 database would have been populated with the tables specified in Project 1B using our dataset.) Make sure your submission is runnable simply after unzipping it. ***Your site cannot assume that any additional auxiliary table or tuples in the CS143 database.*** In previous years, some students wasted their entire 7 minutes just trying to "setup" their Web site, leaving no time to actually test their sites. It is your responsibility to demonstrate all requirements within your allocated 7 minutes. If you fail to demonstrate some parts of the requirements, you will get points deducted for those parts. To avoid this unfortunate possibility, you may want to think about how you can demonstrate each requirement in the minimal amount of time before you come.
2. The TA will check your Web sites using the code you submitted electronically and an existing VMWare image, so you do not need to bring any laptop.