NYU Tandon School of Engineering April 25, 2022
Computer Science and Engineering
CS6083, Spring 2022

Project #2 (due around May 15)

In the second part of the project, you need to create a web-based user interface for the database designed in the first project. In particular, the interface should allow users to register, login, and edit their profile information. They should be able to post questions, select a topic for their posted questions, and select a best answer to their question, to provide answers to questions posted by others, and to give ratings to answers provided by others. They should also be able to see a list of the questions they have posted, a list of the answers they have provided, or any new answers that were provided by others to the user's questions.

Users should also be able to browse and search the site. Browsing should be supported via the multi-level hierarchy, so that a user could click on a topic and maybe get a list of recently posted questions (or questions with new answers) for that topic in reverse chronological order. Searching means that the user could type in a keyword query and get back questions or answers that match the query or score high on the query. You may also consider providing various options to combine browsing and search (say, allowing a keyword search restricted to a particular topic), and offer options such as display in order of relevance versus in reverse chronological order, and you may want to allow users to choose if results of keyword searches should be questions or answers, or both. It is up to you to decide how to support browsing and search – think about what would be most useful to the user.

Note that you have more freedom in this second project to design your own system. You still have to follow the basic guidelines, but you can choose the actual look and feel of the site, and offer other features that you find useful. In general, design an overall nice and functional system. If you are doing the project as a group of two, note that both students must attend the demo and know ALL details of the design. So work together with your partner, not separately. Start by revising your design from the first project as needed. In general, part of the credit for this project will be given for revising and improving the design you did in the first project if necessary.

The followings are a few suggestions for the interface you might want to build for this project. After users log in, they may come to an initial page that might show recent relevant activity by the user, say the questions they have posted or answers they have provided in reverse chronological order. You may want to allow users to edit their questions and answers, but this is not required. Starting from this initial page, you can then enable the other functionalities.

Users should be able to perform all operations via a standard web browser. This should be implemented by writing a program that is called by a web server, connects to your database, then calls appropriate stored procedures that you have defined in the database (or maybe send queries), and finally returns the results as a web page. You can implement the interface in different ways. You may use programming languages and frameworks such as PHP, Java, Ruby on Rails, Python, NodeJS, or Golang, to connect to your backend database. Contact the TAs for technical questions. The main restriction is that the backend should be a relational DBMS with the schema you designed in the first part, with suitable improvements as needed.

For full credit, your interface must take appropriate measures to guard against SQL injection and cross-site scripting attacks. To prevent SQL injection, you may use stored procedures and prepared statements (if your programming language supports them). If your language does not support prepared statements, your code should check and sanitize inputs from users before concatenating them into query strings. To guard against cross-site scripting, outputs to be returned to user's browsers should be checked or sanitized to avoid scripts. Some languages provide functions, such as htmlspecialchars in PHP, to help with this. You should also define appropriate transactions to make sure that multiple users can use the site at the same time. Make sure to address these requirements (protection against SQL injections etc., and concurrency) in your final document, and to describe your solutions.

Every group is expected to demo their project to one of the TAs at the end of the semester. If you use your own installation, make sure you can access this during the demo. One popular choice is to use a local web server, database, and browser on your laptop (in this case, your project can just run locally on your laptop). Also, one thing to consider is how to keep state for a user session and how to assign URLs to content – it might be desirable if users could bookmark a page for a question or answer, or the result page of a search or browsing step that was performed. Grading will be done on the entire project based on what features are supported, how convenient the system is, your project description and documentation (important!), and the appropriateness of your design in terms of overall architecture and use of the DBMS. Make sure to input some interesting data so you can give a good demo.

Describe and document your design. Log some sessions with your system. You should also be able to show and explain your source code during the demo. The documentation should consist of 15 to 20 pages of carefully written text describing and justifying your design and the decisions you made during the implementation and describing how a user should use your system. Note that your documentation and other materials should cover both Projects 1 and 2, so you should modify and extend your materials from the first project appropriately. There will be opportunity to get extra credit by implementing cool extra features, but extra credit is limited to about 5-10% and the TAs will decide what is cool. (Hint: coming up with simple but powerful ways to support searching and browsing, as suggested above, could be one way to get extra points.) There may also be extra credit of up to 5% for doing an early demo before the deadline.