Polytechnic Institute of NYU
Computer Science and Engineering
CS 6083, Fall 2014

November 19, 2014

Project #2

In the second project, you have to create a web-based user interface for the database designed in the first project. In particular, users should be able to register, create a profile, log in, become fans of bands, follow other users, post information about new bands and upcoming concerts, like different genres, RSVP to concerts, rate and review concerts, make lists of recommended upcoming concerts, etc. Additionally, artists/bands should also be able to register, create a profile (have a short bio, list the type of music they play), log in, post information about their upcoming performances, including time and date, venue (location), and ticket prices and availabilities. Since artists/bands can be verified by the company, there should be a visual difference in pages/post by artists/bands vs users.

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 and implement an overall nice and functional system. There will be some extra points available for a nice and smooth design. If you are doing the project in a group, note that both students have to attend the demo and know ALL details of the design. So work together with your partner, not separately. Also, slightly more will be expected if you are working in a team. 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.

A note about the interface you are expected to build for this project. When users log in, they should probably come to a starting page where they can see a feed of posts that may be of interest. For example, they might see posts from artists/bands/users they follow about upcoming concerts, ratings/reviews of past concerts by the users they follow or about bands they follow, other activity by the users they follow, and users/artist/bands they might like based on their previous activity (Recommender system). Also, posting, following, liking, rating and reviewing could be done via appropriate buttons for each entry, but details are up to you. Users should also be able to browse the site (visit other band/user's page, browse different genres and bands that play those genre, etc.), and there should be some way to search using keywords or genre or time or location, for example retrieving all performances that refer to a particular location and/or a particular genre, all performances on a particular day/date-range (say upcoming weekend), or all posts or comments/reviews that contain certain keywords.

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 several different ways. You may use frameworks such as PHP, Java, Ruby on Rails, or VB to connect to your backend database. Contact the TAs for technical questions.

Your interface must take appropriate measures to guard against SQL injection and cross-site scripting attacks. To prevent SQL injection you can 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/sanitized to avoid scripts. Some languages provide functions, such as PHP's htmlspecialchars, to help with this.

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, which means you need to bring your own laptop to the demo. (In this case, your project does not have to be available on the public Internet; it is enough to have it 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 band/artist's page, a user's profile, or the results of a search. Grading will be done on the entire project based on what features are supported, how attractive and convenient the system is for users, your project description and documentation (important), and the appropriateness of your design in terms of overall architecture and use of the database system. 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. Bring your description (documentation) and the logs in hardcopy to the demo. You should also be able to show your source code during the demo. The documentation should consist of 10□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 nice extra features, but available extra credit is limited to about 15%. There may also be extra credit of up to 5% for doing an early demo, before the deadline.