

Project #2 (due December 15)

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, post notes, and comment on notes they have received. Users should be able to define filters for receiving notes and be able to send and answer friend requests.

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 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(s) you are expected to build for this project. There are really two parts. One is the interface that a mobile user would use (though based on a browser and not an app), which allows users to log in, define filters, post notes, etc. The other part is a very simple interface that allows a user to choose the current time and current location – this is needed so that you can demo your system to the TAs without running through Brooklyn for several hours. This second system might have a slider to select the time, and allow users to click on a map (via Google Maps or similar) to select their current location, and maybe should also display ALL notes on the map (not just those currently visible to this user), to make it easier for the TAs to explore the data and test the system. You may want to put this second interface below or next to the main interface on every page, to simplify the demo.

Users should be able to perform all operations via a standard web browser, unless you have decided to build an app instead. (But if you do build an app, make sure it can be demoed to the TAs in a reasonable way.) 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 frameworks such as PHP, Java, Ruby on Rails, or VB to connect to your backend database. Contact the TAs for technical questions. The main restriction is that the backend should be a relational database based on the schema you designed in the first part, with suitable improvements as needed.

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/sanitized to avoid scripts. Some languages provide functions, such as PHP's htmlspecialchars, 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 writeup.

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 you then bring to the demo. (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 with a job or a company description, or the result page of a search that was performed. 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 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. Bring your description and logs to the demo (softcopy is OK). You should also be able to show and explain your source code during the demo. The documentation should consist of 15–20 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 extra features, but extra credit is limited to about 5–10%. There may also be extra extra credit of up to 5% for doing an early demo, before the deadline.