1. A "design document" for your project in the form of a file called `design.html`, `design.pdf`, `design.php`, or `design.txt` that discusses, technically, how you implemented your project and why you made the design decisions you did. Your design document should be at least several paragraphs in length. Whereas your documentation is meant to be a user's manual, consider your design document your opportunity to give the staff a technical tour of your project underneath its hood.

We will first address our most basic design decisions with regards to the functionality of the website. For one thing, the reason that database registration requires a username and password (despite the fact that there is currently no log-in function) is because, though repeatedly accessible accounts are unnecessary for the scope of what we sought to accomplish for CS50 (primarily providing a platform for matching organizations), we do plan to continue developing this website after the fair, and eventually, as SAN grows and gains new members, we would like to give those members more powers via the website.

Though in our original goals, we planned to e-mail members each time an organization with the skills they wanted joined the database, we ultimately decided this was a bad idea for several reasons. One, people are often frustrated by frequent e-mails, and we did not want to "spam" users. Two, needs change as goals change and so as organizations develop new plans, the skills they require may change. Three, organizations may want to pick their partners based on common or similar interests, not just skillset, and so our original plan might not be especially helpful. For all these reasons, we decided to make the database searchable by all users (rather than e-mailing members to match them up) so that organizations can develop their own search criteria for finding partners and change these criteria as needed.

When you register to become part of the database, and when you search the database, the fields "cause", "skills you need", "skills you have" (if you're registering) and "skills they need" (if you're searching the database) are all dropdown lists, because we wanted to ensure that the values entered would be easily searchable and that organizations would not miss out on potential partners because they searched for a synonym instead of the exact phrase that the other organization entered. Because we realize that a dropdown list can feel limiting, we added a component at the bottom of both pages, suggesting that users e-mail the webmaster if they feel more skills need to be added to the list.

The reason that we limited the number of skills organizations could claim to have or be looking for, is because we wanted to be sure people listed on those areas in which they truly excel and can help others, rather than finding ourselves with a database full of users who have treated the process like a LinkedIn account and listed hordes of skills they are not truly proficient in. Additionally, we were hoping to make groups hone in on the most essential skills they lack to accomplish their goals, rather than mindlessly registering as needing many skills, when their goals only require a couple.

We chose to create a form.css file so that our pages would be consistent. For example, the "add event" functionality on the calendar page, the registration form, and the "working for SAN" application form, all use form.css to keep them uniform and professional. We decided to link css pages as needed instead of stuffing information into the header because this decision allowed us to minimize overlap between pages, which might have similarly named divs that could cause the css to become confusing or malfunction. This decision also improved efficiency by reducing the number of files that needed to be loaded for any given page to load.