Kyle Shamblin

7/25/13

Intro to Internet Computing

Oge Marques

MP2: School Information Form

For the second mini-project I decided to create a form that an incoming student might see. The form is made up of four components: The Basic Information, the Personal Information, the Class Picker and the Summary of the whole thing. The target audience is anyone entering college for the first time, specifically for classes at Reed College (the College I used for the classes picker). The first page contains a form for basic information such as first name, last name, etc. The second page is supposed to be a simplified version of a financial aid form. I didn’t want to use too many fields so I made it simple. The third page allows the user to select his/her classes for the next semester. The information for the class is loaded once the time for each class is selected. The final page is a summary of all of the information given. The user is able to go back and forth between pages and re-edit their information. The project can be found [Here.](http://lamp.cse.fau.edu/~kshambli/MP2/)

There is only one page in the entire site: index.php. Everything is redirected through here. If the user wishes to fill out any information on the next page they must have already filled out information on the previous pages. Every page is defined by a variable called page which is defined in the URL.

The architecture of the code works in a separated way. Each page has its own file and Javascript file. There is also a template file that each page is loaded into. There’s one CSS file for all of the pages.

The code works in a very simple way. First of all, every page cannot be seen directly because of a security I implemented which defines a constant variable on the main page and if this variable is not defined on every other page then the user does not gain access. A couple of constants are defined that help me throughout my code as well as several helper functions. Once this is all done the session is started which is what is mainly used to save the information stored by the user. It’s also what allows the user to go back and edit the information.

The next area of code I’d like to discuss is the page setup. First off, if the page set in the URL is not valid then the page defaults to 1. If the page was never set at all in the URL then the session is destroyed and the page is set to the first page. Once the page is finally set the code checks to see if there is any information in the session regarding that page. This information will then be inserted into the input fields on the page. The template for the site is then set as well as the template for the page and the title of the page.

After this is all set the application checks to see if any information has been posted yet. If it has, based on the page, the application validates the passed information. If there are no errors, the page is forwarded to the next page because the current page has completed its journey. If there are errors, the errors are printed to the screen and the user has to correct the information. Finally the page is printed to the screen.

The first two pages are fairly simple, using the basic html form features and some JQuery for validation. The third page is what I worked on mostly. It’s a class adder. When the page loads, before the user can do anything, all of the classes are loaded from an xml file into JQuery. I grabbed the xml file from [Here](http://www.cs.washington.edu/research/xmldatasets/www/repository.html). Once the page loads the user can add a class. Once the class is loaded the code finds all of the classes with that particular title and grabs the times for them. The user then picks a time and the class information is loaded on the left. The user is able to dynamically add and remove classes. I set the max to four classes because I don’t want them overloaded with school work. Realistically they could add as many classes as they want if not for that limit.