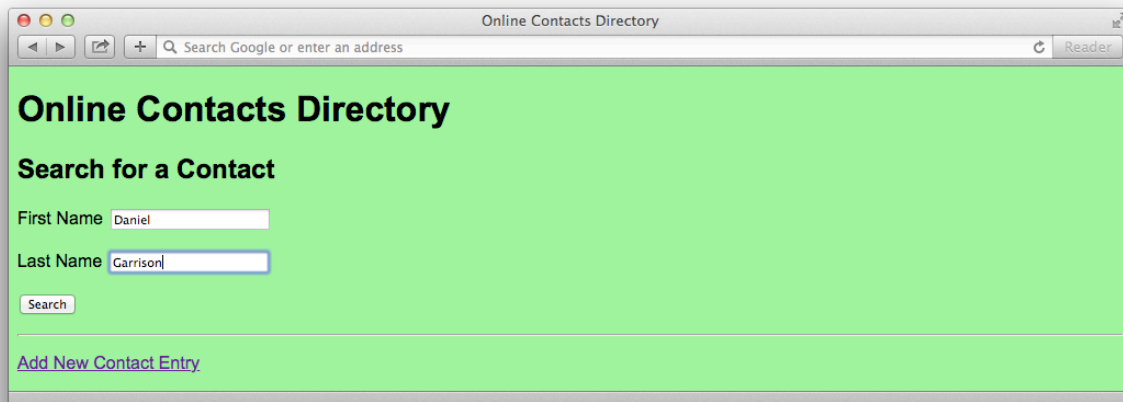**Online Contacts Directory**

For this assignment you need to design and implement an online contacts directory that can be used to retrieve different types of contact information including phone number(s) and address data. This directory may prove beneficial as once it is available on the web, you can access this information from any net enabled device. In addition, as you may get new or updated information on a contact while away from your home computer, having the ability to add new contacts or update existing contacts wherever a net enabled is available may also prove beneficial. Similar applications to this are Google Contacts, iCloud Contacts, and Microsoft Office 365.

Design Requirements

You should create an online contacts directory application that saves entries to a single text file. Include standard contact directory fields in the text file such as *first name, last name, email address, phone number, street address, city, state, and zip code*. You will want to plan how to save this data to the file for easy retrieval. Hint: Your textbook may provide helpful insights.

As part of your application, one web page will serve as a main "directory," where you can search and retrieve records along with a document that allows you to add contacts. You should have another web page that you can use to edit contacts. All pages should allow the user to return to the main "directory".

Below is an example of the main "directory" output you should have before formatting:

Whether a user is searching, adding a new entry, or updating an existing contact you should verify that they have entered a value in each field. If the have not, then you should indicate this to them via server-side scripting. When an entry has been successfully saved or updated then you should tell the user. If a search, save, or update was not successful then you should indicate this to the user. You should also alert the user if the text file containing the contacts does not exist.

On the next page is an example of one of the error outputs you should have before formatting:

You must enter a value in each field. Click your browser's Back button to return to the form.

Return to Directory
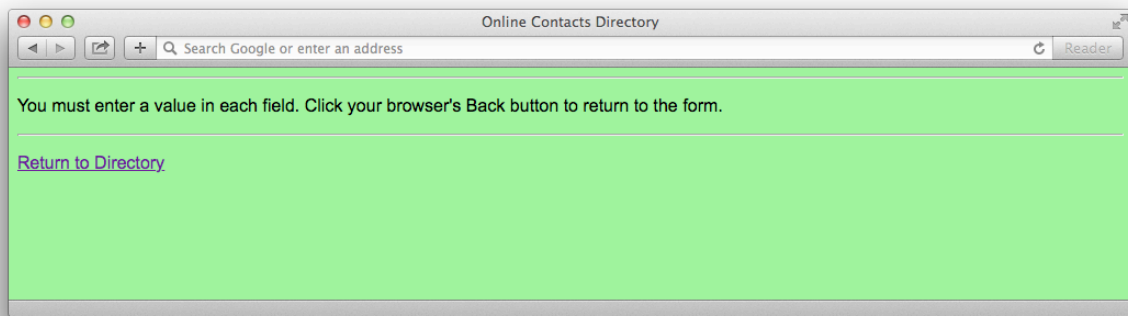
On each entry form, the state field must be set up using a <select> drop-down list as shown in the example diagram. In addition, the update page must pre-select the state previously saved along with pre-filling all the other fields with the values previously saved.

Below is an example of the edit entry and add entry output you should have before formatting:

First Name Daniel   Last Name Garrison

Email Address dgarriso@gmu.edu

Phone Number 856-555-1212

Address 12 West Ave   City Westover

State Alabama   Zip NJ

Update Entry

Return to Directory

New Contact Entry

First Name Grace   Last Name Wu

Email Address gwu@gmu.edu

Phone Number 703-555-1212

Address 4400 University Dr   City Fairfax

State Virginia   Zip 22030

Add Entry

Whenever you are writing data to the file, you should appropriately lock the file to prevent a mismatch of data being saved. If the file cannot be locked then you should alert the user to retry submitting the form in

a few moments as other updates are being performed. Constants should be used for values that do not change, and variables for values that change during the course of execution. You should include Last Modified Date information on your page. This should be provided dynamically by using the built-in `date()` function within the `getlastmod()` function.

At the point of final submission, your application should be laid out in the *Lab Content* area of your web site. When laying out your web page, you should retain the common navigation and structural components (header and footer) of your website even though not shown in the image. This will require you to incorporate the SSI file references in this lab assignment as you did on previous the lab assignment. SSI files should not be moved but rather you should provide the path to their original file location. The common navigation and structural components should present visual continuity across all the web pages of your website and should not shift at all when going from web page to web page. In addition, while the image does not demonstrate this, you need to include a working link on each page to get back to your home page. Each page of the assignment should load into the *Lab Content* area only.

For the main "directory" page, you should use a file name that will automatically load when a user specifies only the folder name in the URL. The file should be uploaded to your Lab Assignment 3 directory. Your XHTML file should be well-formed and created using a strict XHTML DTD. You can create the web page using any text editor (Microsoft Notepad, RogSoft Notepad+, Adobe Homesite, etc) that is not WYSIWYG capable (Adobe Dreamweaver, Microsoft Word, etc).

You should use XHTML for the content of all pages and CSS for the stylizing and positioning of each of the pages. Using an external .css file, add formatting of your choice to make the output more aesthetically pleasing. These styles should be implemented as part of the early submission points. While you should continue to use the external CSS you created for your home page, you should not modify or update the file. Your new CSS file should not conflict or modify the layout and positioning of earlier lab assignments.

Approach
For the first submission point, create a simple XHTML page for all the pages of the online contact directory. These pages contain no PHP however some of them will have forms that submit to a separate .php script files. Create a text file with starting records that correspond to the order of the form fields for contact names, addresses, etc.

After creating the non-PHP XHTML pages, you may want to start by determining, coding, and validating the .php files using XHTML (and absent of any CSS). You can simply assign a record to an array for early functional testing. As part of this, and after the first submission point, begin to populate these arrays with the contents of the file instead. Add the ability to write to the file. Once working like the output given, you can begin adding the CSS formatting. Finally, validate all the pages and correct any ill-formed tags where necessary.

Submission
You will need to upload these web/script pages and CSS file to your Lab Assignment 3 directory within your Zeus/Helios server space, while keeping the original SSI and CSS files in their original file location. Verify that all the uploaded pages work in the browser, as this is what will be used for grading. Also for each submission point, you must submit all the files you create to the appropriate Assignment drop box on Blackboard.

Notables
▪ Remember that the lab assignment must clearly display your name in the interface (if any) and be commented with your name in all files.
▪ Try to be consistent in the naming of your files as this will help you later on.
▪ Feel free to use your book and notes for general information on these technologies.
▪ If need be, for reference material on HTML, you may want to visit http://w3schools.com/html/ which we discussed in class.

- For reference material on CSS, you may want to visit http://w3schools.com/css/ which we discussed in class.
- For reference material on PHP, you may want to visit http://php.net which we discussed in class.
- Websites other than those provided in this lab should not be utilized.
- Just a reminder… this is individual work so while you can provide general assistance to other students while in the lab class, you should not work together or give solutions. Assistance provided to others while outside the lab class should only be through the use of the discussion forums.