COMP 3512
Assignment #1: Data-Driven PHP *(version 2a)*

Due Saturday October 21, 2016 at midnightish
Changes in yellow

Overview

This assignment provides an opportunity for you to demonstrate your ability to generate dynamic web pages from database content using PHP. In this assignment you will be working with the Book case study from the textbook.

Beginning

Create a new public workspace named ***yourlogin*-comp3512-assign1** (e.g., fsmi9876-comp3512-assign1). In the Clone from Git field, enter https://github.com/rconnolly/comp3512-f2017-assign1.git and select the PHP template.

We will be using the **book-complete.sql** file to populate our database. The easiest way to do so is to use the Terminal. Once you have started MySql (type **mysql-ctl cli**), enter the following commands:

```
create database book;
source book-complete.sql;
```

There are some 20000+ records to import so it will take about 5 min to run. After it completes, run a few test queries, for instance, by entering the following commands:

```
use book;
select * from Imprints;
```

Because MySql maps table names to file system files and because Linux-based operating systems are case sensitive, MySQL table names on cloud9 are also case sensitive. You will only have to import this data once.

Submitting

For me to mark your assignment, you will need to share your workspace with me once it is complete. To do so, click the Share button (it has a gear icon) in the upper-right part of the Cloud9 workspace. Invite me via my email (rconnolly@mtroyal.ca) and be sure to give me RW access (otherwise I can't run anything).

Grading

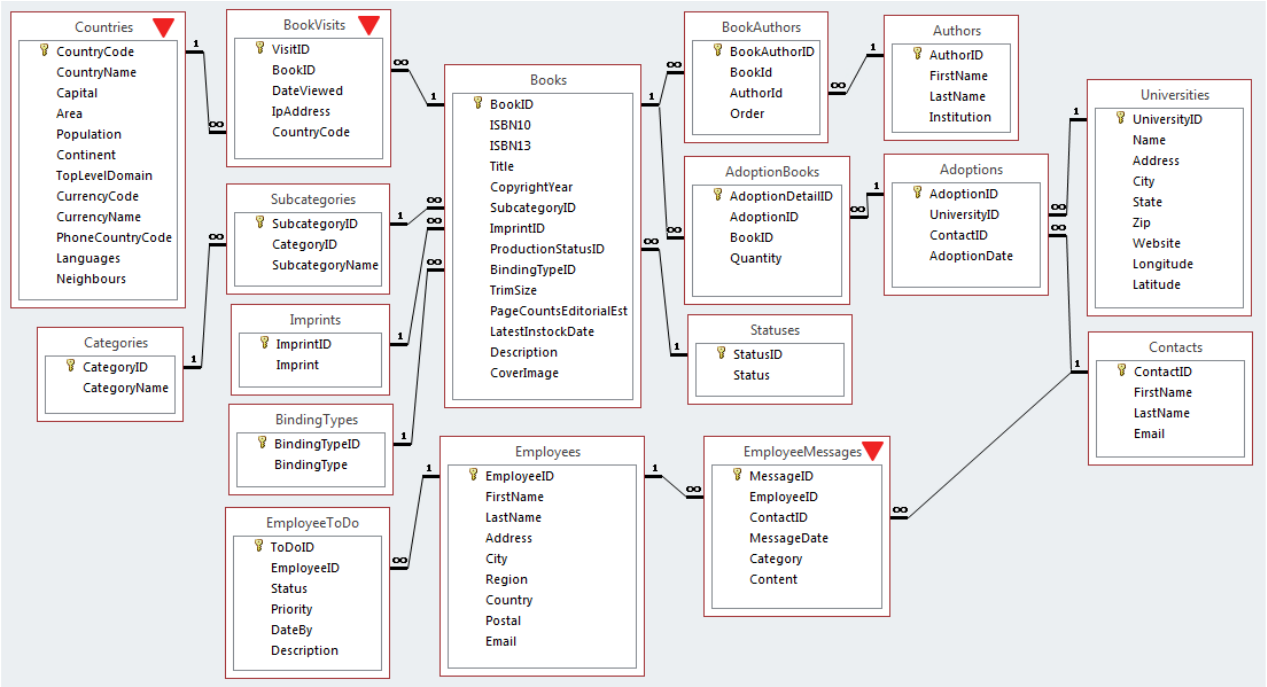
The grade for this assignment will be broken down as follows:

Visual Design and Styling	15%
Programming Design	20%
Functionality (follows requirements)	65%

Data Files

You have been provided with the necessary SQL script that you will need to import into MySQL (see instructions on first page). A Microsoft Access version of the database has been supplied as well in

case you need to want to test run queries or browse the data outside of Cloud9. Images for the books have been provided as well.



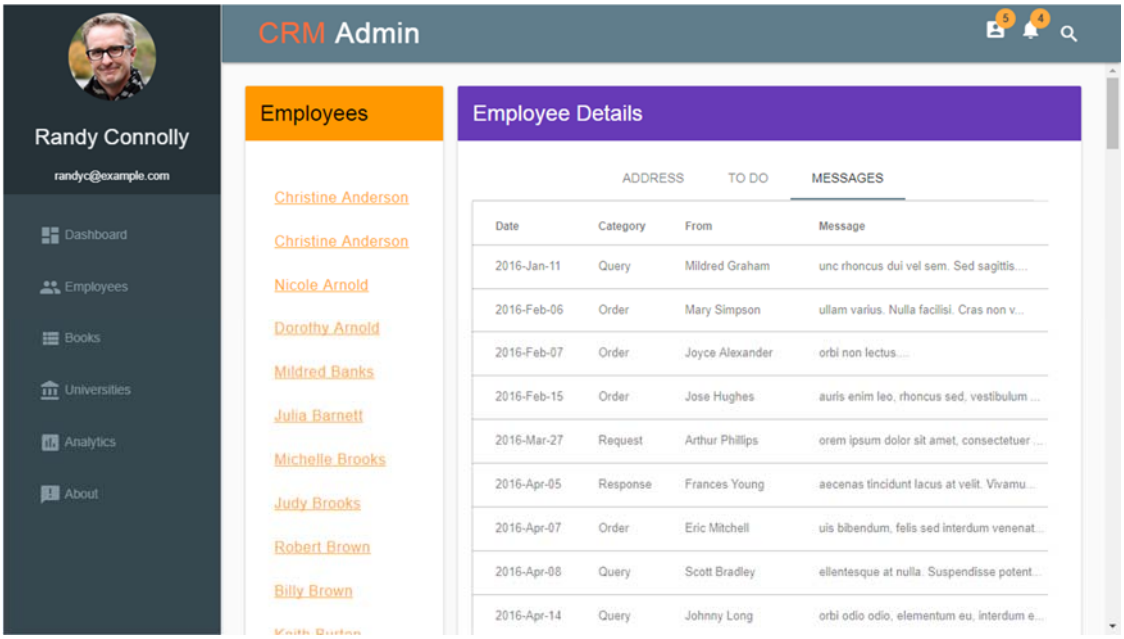
Requirements

This assignment consists of 5 pages with the following functionality:

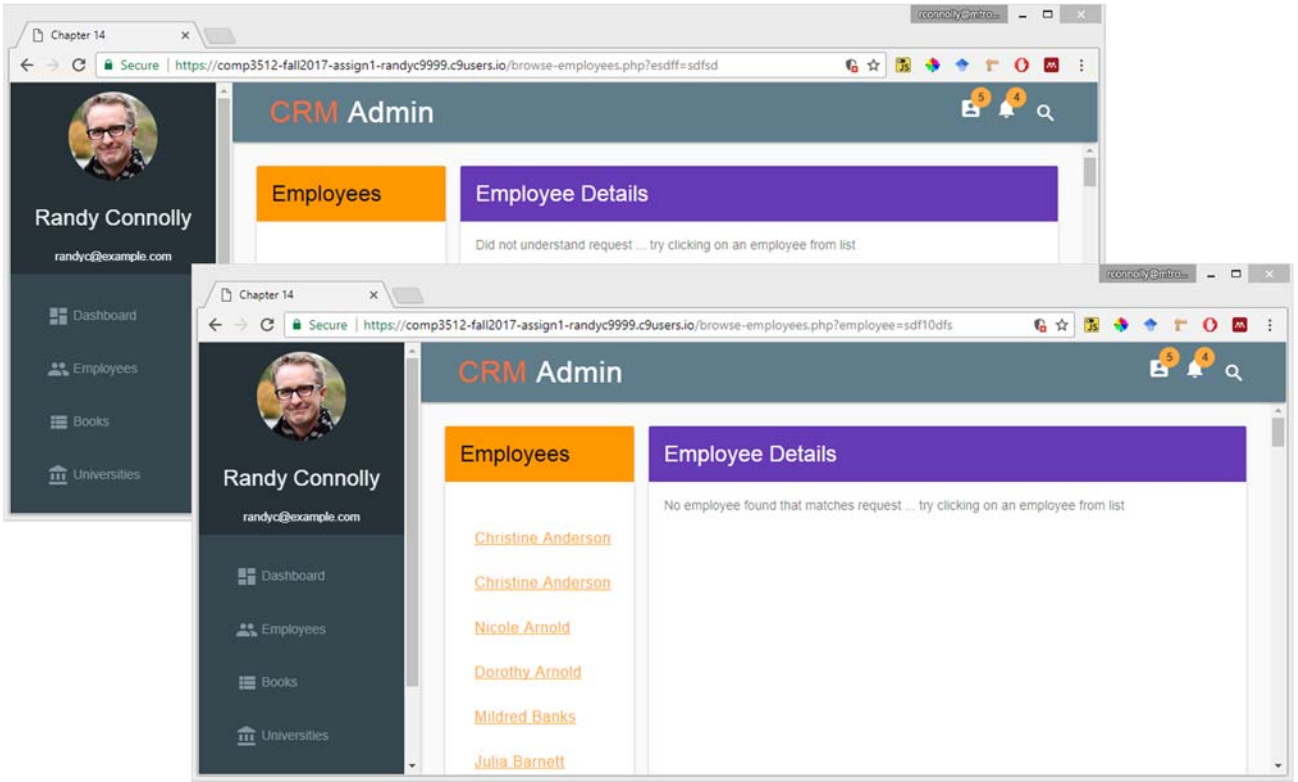
1. This assignment makes use of the Google Materials Lite (MDL) CSS Framework (https://getmdl.io/) to style and layout your pages. Like with Bootstrap, this framework does have a little bit of a learning curve, but since you will be building up from the lab exercises, you will likely be able to get started via copying and pasting. In the second assignment, you will port your application to use the newer Google Material Components (MDC) for the Web CSS Framework (https://material.io/components/web/). If you want, you are welcome to use the newer framework now on this assignment, though it will be slightly easier to stick with MDL since the end-of-chapter projects already use it.
2. The site **must** use a single CSS file for any additional text and color formatting. The design mark will mainly be judged based on how sensibly you have used the MDL framework in the `browse-universities.php`, `browse-books.php`, and `single-book.php` pages.
3. All pages **must** have six links: to `index.php`, `aboutus.php`, `browse-universities.php`, `browse-employees.php` , `browse-books.php` , and `analytics.php` pages. Modify the list of links on left-side of all your pages. Change the name of the header individual from John Locke to your own name and the picture to one of yourself (or some other image … just don't use my picture anymore). The analytics page will be completed in a future assignment so this link doesn't have to work.
4. You will need to write database access code for most of your pages in this assignment. The programming design component of the grade [worth 20%] will in large part be determined by how you structure this database access. Remember that the preference is for markup that has minimal programming logic within it *and* for programming without tons of echoed markup.

   I would also strongly recommend creating the following pages in the order provided below.
5. There must be a page named `index.php` and must be the destination for the Dashboard link in your site. The text and links for the images on the home page can be hard-coded. This page will contain the following links formatting as MDL (Material Design Lite) Cards: Browse Universities, Browse Books, Browse Employees, and About with links to the appropriate pages. It might be nice to make the cards include an image.
6. There must be a page named `aboutus.php`. It should have your name, the course name and number, date, and anything else you'd like to put here. Somewhere on this page, provide a list of all resources you are using that you did not create (e.g. MDL, images, etc). Try to make it look nice and make it fit with MDL (or MDC) styles.

7. There must be a page named `browse-employees.php`. It must display a list of employees from the Employees table (sorted by last name) as a list of links. Each employee name will be a link back to the same `browse-employees.php` page but with a query string parameter containing the employee id. When the page receives a request with this id, then display a separate MDL card containing a set of tabs: a tab group containing the address information for employee, a tab group containing the employee to-do list in a table, and a tab group for employee messages also contained in a table. This is essentially an expanded version of Chapter 14, Project 1. In the employee messages tab, display a table including the date, category, from (contact first name and last name) and the first 40 characters of the message.



This page (and the next three) should be able to handle a missing and/or incorrect query string.



8. There must be a page named `browse-universities.php`. Like the `browse-employees` page it should display a list of university names sorted by title. Because there are over 600 it should only display the top/first 20. The page should also contain a drop down list of US state names (sorted alphabetically) along with a button beside the filter list. Our book database doesn't contain a table of US Sate names, so you will have to create and populate this yourself (see http://kimbriggs.com/computer/mysql-create-us-state-table-script for the relevant MySQL script). I have also added one to the GitHub repo. You can manually add this yourself if it wasn't available when you started your workspace.

   Once the user clicks on the filter, it should perform the filter; that is, re-request the page and update the university list to display only those universities from the requested state (still only the first 20). Be sure to make the first item in the list an option to remove the filter (i.e., see the first 20 universities from all states).

   Each university name should be a link back to the `browse-universities.php` page with the selected university id as a query string parameter. Like with the `browse-employees` page, once a university link has been clicked, the page will display the details (name, address, city, state, zip, website, latitude, and longitude) about the university in a separate card on the same page.

9. There must be a page named `browse-books.php`. This page displays a list of multiple books sorted by the title. The list should contain a thumbnail of the cover, the title, the year, subcategory name, and imprint name. Because there are hundreds of books, only show the top 20. Each of the thumbnails and titles must be links with the ISBN10 field as the query string to the `single-book.php` page (see below).

   Initially, this page must display all the books in the books table. The user should be able to filter the list by specifying the subcategory or imprint in two lists of links, populated from the subcategory (sorted by name), and imprints (sorted by name) tables. Each list should be in its own card. As with the unfiltered list, only display the top 20 matches for the filter. The first item in each list should be All Imprints/Subcategories: if the user clicks on this link, it should perform the filter. Be sure to make the first item in the lists an option to remove the filter (i.e., see all the subcategories/imprints).

10. There must be a page named `single-book.php`. It should display the details for a single book specified by the ISBN10 value passed in as a query string. This should include a larger version of the cover as well as the following information: ISBN10, ISBN13, Title, CopyrightYear, SubCategory, Imprint, Production Status, Binding Type, Trim Size, Page Count, and Description. Don't display the foreign keys; display the relevant name. For instance, you wouldn't want to display the SubcategoryID field value of 16; instead you would want to display its related name ("Principles of Economics") from the Subcategory table. This must be accomplished with a single query with multiple inner joins. All of this information should be contained within a single MDL Card element. This page must include two other cards. Each of these will require separate queries. One of these will contain a list of authors for the book, sorted by the Order field. The other card will display a list of universities that have adopted the book.