ITEC 3280 Web Programming – Final Project

In this project, you will create a dynamic, data-driven faculty/staff directory data management web app using HTML5, CSS, JavaScript, PHP, and MySQL database. The application must incorporate best practices and standards for a Web-based information system.

Part I – MySQL Database (20pts)

By using *phpMyAdmin* in the XAMPP, create a MySQL database (named '*directorydb*') with the following tables. The last step in the Final Project is to Export your database, complete with test/sample data, so that I can use it to test your application. Don't forget your Export at the end!

1. **Departments**

The *Departments* table contains information about the different departments at the University.

Department ID (INT, PK, auto-incrementing, *deptID*)
Name (VARCHAR(50), *deptName*)
Phone Number (VARCHAR(20), *deptPhone*)
Email (VARCHAR(50), *deptEmail*)
Office Location (VARCHAR(100), *deptOffice*)

Then add some fictitious data to populate the table. Just a few records will do.

2. **Persons**

The *persons* table contains properties of the faculty/staff member in the directory. Faculty/Staff are assigned to one department.

Person ID (INT, PK, auto-incrementing, *persID*)
Email Address (VARCHAR(50), persE*mail*)
Password (VARCHAR (50), persP*assword*)
First Name (VARCHAR(50), *persFName*)
Last Name (VARCHAR(50), *persLName*)
Phone Number (VARCHAR(20), persPhone)
Office Location (VARCHAR(100), persOffice)
Department (INT, persDept)

Then add some fictitious data. The *persDept* field in *Persons* table will be an integer value from sample department data you added in step one. If you use an integer value that cannot map to the *Departments* table you may have problems below. Add at least 5 people to your directory.

Part II – Web Development (HTML and CSS) (20 pts)

This web application will consist of four web pages. Each page will need to meet the following requirements:

1. Research and determine optimized web page layout dimensions based upon current browser usage statistics. The layout dimensions should be used on each page.

2. Use CSS techniques (no tables or frames) to create a page layout template. This template should be used for each page.
3. The document must conform to HTML 5 and CSS standards. Use the W3C validator! (http://validator.w3.org/) and the CSS validator (http://jigsaw.w3.org/css-validator/)
4. Site must be tested for accessibility issues to conform to section 508 and W3C accessibility guidelines using the instructor's specified tool.

Part III - Web Scripting (PHP and JavaScript) (60 pts)

Create the following pages based on the HTML/CSS template created in part II. Add the following PHP/JavaScript functionality:

5. **login.php**

   Create a login page that requires the user to input an email address and password. When the "Login" button is clicked, the input is validated against the values stored in the "*persons*" table. If the email and password is incorrect, error messages are displayed on the page. If the email and password are correct, the user is redirected to *dashboard.php*. In addition, the login page contains a 'register' link so that a new user can register. If the register link is clicked, the user is redirected to *register.php*.

6. **dashboard.php**

   Create a *dashboard.php* page that generates 2 lists of the following information using appropriate HTML controls; the report should show all the fields in the database for each entity.

   1. Listing of all the Departments w/ an Edit button/link for each that, when clicked, goes to *department.php*. Pass the *deptID* value via querystring for which was clicked for later use.
   2. Listing of all the People in the Directory with their respective department names in plain-text. An SQL INNER JOIN keyword will help you draw the relationship through *persDept* and *deptID*. https://www.w3schools.com/sql/sql_join_inner.asp

7. **register.php**

   Create a new faculty/staff register page that contains appropriate HTML form controls to allow the user to input the following user data. Each input should have an associated <label> tag.

   Email Address - textbox
   Password – textbox with *type="password"*
   First Name - textbox
   Last Name – textbox
   Phone Number – textbox
   Office Location –textbox
   Department – dropdown list (<select>). The value of the <option> tag should be the *deptID* and the user should see the plain-text from *deptName*. These should be generated via the database and not hard-coded in.

Submit Button

When the submit button is clicked, a **javascript** function is called to validate the user input. The **javascript validation routine** should ensure that all fields are required/input, password is at least 8 characters in length, and email should contain the '@' and at least one dot ('.'). (You may use the following simple email regular expression, /^\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*$/).

If all data is correct, the data is inserted into the '*persons*' table. If any input is incorrect or missing, error messages should be displayed within span tags to the right of the form controls. Make it easy for the user to figure out how to complete your form and put data in your database.

8. **department.php**

   The *department.php* page shows the department data associated with the current department passed via querystring from the *dashboard.php* page. The department should be displayed within appropriate form controls to allow the user to edit the information. The page also includes an 'Update' button. If the 'Update' button is clicked, any changes to the data are updated to the '*departments*' table in the database. All fields except the Primary Key (PK) should be editable. If successful, redirect the user back to the dashboard; if unsuccessful, display appropriate errors to the user.

9. The site must utilize unobtrusive coding. All JavaScript and CSS should be coded in an external file .js and .css files. All JavaScript and CSS files should include comments and other appropriate documentation. All JavaScript routines should include appropriate error handling.

10. Export the '*directorydb*' as SQL format and turn it in with other files (submit a zip file). Make sure your export contains the information for both tables and contains your sample data. If I don't see an export 20points from Part I is unsubmitted!