CS 328 - Week 10 Lab Exercise - 2024-03-29

Deadline

Due by the end of lab.

Purpose

To practice using PHP to request data from Oracle, including using it to build a dynamic form widget and using a bind variable to help thwart SQL injection.

How to submit

Submit your files for this lab using `~st10/328submit` on nrs-projects, each time entering a lab number of **90**.

Requirements

- You are required to work in **pairs** for this lab exercise.
  - This means **two** people working at **ONE** computer, one typing ("driving"), one saying what to type ("navigating"),

    while **BOTH** are looking at the **shared** computer screen and **discussing** issues along the way.
- Make sure **BOTH** of your names appear in each file submitted.
- When you are done, before you leave lab, somehow e-mail or copy the lab exercise files so that **BOTH** of you have copies, and **BOTH** of you should submit these files using `~st10/328submit` on nrs-projects, with a lab number of **90**.

The Set-up

Consider the posted example **`328lect10-2-query-empls.php`**.

- It connects to Humboldt's Oracle student database using OCI (in a no-end-user-login approach) and queries for employee information.
- It builds a `table` element containing the queried information --

...but one could reasonably use a `select` statement's results to populate a **`select`/drop-down** element within a form as well.

So -- also consider Week 9 Lecture 2's **`php-form-handling.php`** example, a postback PHP that creates a form as well as its response.

Finally, also consider the posted handout **`328bind-variable-intro.pdf`**, that summarizes the basics of using Oracle/PHP bind variables.

Your Tasks

Create a PHP document named **`328lab10.php`** that **both** creates a form **and** crafts a response to that form when it is submitted (as demonstrated in the posted Week 9 Lecture 2 example `php-form-handling.php`).

- Fill in its opening comment block with the URL I can use to run your version, your name(s), and the last modified date.
  - It will be a penalty if your document does not display using this URL when I/the grader try it.
- Include an appropriate **`title`** element.
- You are welcome and encouraged to use the function **`hum_conn_no_login`** in `328lab10.php`!
  - You can get a copy of this in your current directory on nrs-projects using:

    ```
    cp ~st10/hum_conn_no_login.php .  # remember the space and dot!
    ```

  - Be sure to use **`require_once`** to include this in your **`head`** element, and then you can call `hum_conn_no_login` as desired in your `body` element.
- Include an appropriate **`h1`** element.
- Include **your last name** within a **`p`** element that you add to the CS-328-standard **`footer`** element (the one from `html-template.html`).

What form?

When your PHP is initially called, it should set up a query to Oracle asking for department numbers and department names. It should use them to create a **form** containing at least:

- a `label` logically connected to...
- ...a **`select`**/drop-down widget with **`option`** elements:
  - whose **contents** are those just-queried **department names**
  - whose **`value`** attributes are those just-queried department numbers
- a submit button

What form response?

When its form is submitted using `method="POST"`, your PHP should:

- appropriately **sanitize** the department number submitted by this form
- use it as the value for a **bind variable** within a `select` statement querying for **at least two** attributes from the `dept` table, the `empl` table, or both,
- which it then adds to the response in a pleasing, strict-HTML-style way.

Note that you will lose substantial credit if you use concatenation to include the submitted department number within your `select` statement string -- you are **required** to use a bind variable instead!

For example:

- This could be as simple as querying for that department number's name and location, and then displaying those tastefully within a `p` element.
- This could query for the last names and salaries of employees with that department number, and then display them in a `table` element.
- This could query for the department location for this department number and the last names of employees with that department number, and use the department location in an `h1` element and list the employee names

at that location in `li` elements within a `ul` element.

Reminder: How can one strict-validate the HTML resulting from a PHP?

Beware -- if you put the URL of this PHP directly into the validator at https://html5.validator.nu/, it looks like it does regular-HTML validation (not strict-validation) of just the response containing its form. This is not a bad start, but it is not strict-validation, nor does it check your PHP's response with a submitted form's values.

Here is one approach I have successfully used to strict-validate a multi-document PHP such as `328lab10.php`:

- Put your `.php`'s URL in a browser and view its source, copy and paste that source into a file with suffix `.xhtml`, (for example, `328lab10-1.xhtml`) and put the URL of that `.xhtml` document into the validator.
- Put your `.php`'s URL in a browser and submit its form, then view that *response*'s source, and copy and paste that *response*'s source into a file with suffix `.xhtml`, (for example, `328lab10-2.xhtml`) and put the URL of that `.xhtml` document into the validator.

Optional additions

- You may add an external CSS formatting your PHP document if you would like; if you do so, also submit that `.css` file.

Submit your resulting `328lab10.php` (and all additional files it uses, if any) with a lab number of **90**.

BEFORE you leave lab:

Make sure that you **both** have copies of the file `328lab10.php` (and all additional files it uses, if any), and you BOTH submit these using `~st10/328submit` on nrs-projects, with a lab number of **90**.

How the navigator can get today's lab files:

- Because this happen to be a file the nrs-projects web server has to be able to reach, the navigator should be able to get a copy of these files from the driver using an approach like this:
  - Assume the driver has username `ab12`, and the navigator has username `yz89`.
  - Also assume the driver created, in their `public_html` directory, a sub-directory named `328lab10`, and this sub-directory contains the files to be submitted for the lab exercise.
    - The NAVIGATOR `yz89` can now:
      - log in to THEIR nrs-projects account

```
cd public_html
mkdir 328lab10   # or other name they choose
chmod 711 328lab10
cp ~ab12/public_html/328lab10/328lab10.php . # note space & dot!
```

      - And now the navigator `yz89` has their own copy of this file.
- I will leave it up to the navigator to decide if they would like to UPDATE their `l328lab10.php` so their opening comment includes the URL to *their* copy, or if they want to leave the URL for the driver's copy.

- HOWEVER: remember that you *will* lose some credit if this URL does not work when I or the grader paste your submitted file's URL into a browser, in either case.