COMPX222 – Assignment 3:
Database Driven CRUD Application (30%)

Deadline: Friday 21st June by 23:59 (at the latest)

Task

In this assignment, you will build a web application for storing and retrieving people's contact information. This will be a database driven CRUD application, allowing users to Create, Read, Update and Delete contacts from the database.

The application will have:

- A page for displaying all the contacts stored in the database
- A page for displaying detailed information about individual contacts.
- A form enabling users to input new contacts, including details such as name, phone, email and address.
- An additional form allowing users to modify the details of existing contacts.
- Functionality that enables users to delete a contact from the database.

To achieve this, you will use PHP to dynamically generate page content, validate the form data, and interact with the database.

You are free to create any design you want for your application; however, ensure that it meets all of the requirements specified below.

Goal

The goal of this assignment is to demonstrate your ability to use server-side PHP scripts to dynamically generate content, process and validate HTML form data, and interact with a MySQL database. You are also expected to demonstrate your ability to build a responsive web application with a design that is both user-friendly and consistent across all pages.

Requirements

1. Your application must consist of the following pages: index.php, details.php, add.php, update.php.
2. The **index.php** page serves as the homepage and should display all contacts in **alphabetical order**. You do not need to show all of the contact information for each contact; you may choose to show the name, email and phone number, but not the address. Clicking on a contact should redirect the user to the details page for that contact.
3. The **details.php** page displays a single contact and should display all of the information for that contact. This page also includes a delete button and a link to the update page, where that contact can be edited.

4. The **add.php** page contains a form where users can add a new contact. This form should be a POST request and it should contain at least the following inputs: Name, Phone, Email, Address. After the form has been submitted and the contact successfully added, the page should redirect to user to either the homepage or the details page for that contact.
5. The **update.php** page contains a form where users can update the details of an existing contact. This form should also be a POST request and it should be possible to update any of the fields, e.g. Name, Phone, Email, Address. When the user visits this page, the inputs should be pre-populated with the data for that particular contact. For example, the Name input should already contain the name of the contact. To achieve this, you will first need to retrieve the contact from the database using its ID. After the form has been submitted and the contact successfully added, the page should redirect the user to either the homepage or the details page for that contact.
6. Form **data must be validated with PHP**. Invalid data should be reported with appropriate error messages, styled to be recognizable as errors. This validation should only occur when the user submits the form, not when moving between inputs.
7. The following **validation should be applied to both the add and the update form**:
    a. The Name field cannot be empty.
    b. Either the Phone, Email, or both must be provided; leaving both empty should trigger a validation error.
8. Users **must be able to delete a contact** from the database. This can be achieved via a Delete button on the details page. After a contact has been successfully deleted, the user should be redirected to the homepage.
9. All pages of your web application **must be responsive**. This means they should usable on different screen sizes and devices. You can use CSS grid, flexbox and media queries to achieve this.
10. All pages should have a **common header and footer**. You should implement this by creating a single header.php and footer.php file and using PHP includes to include them on every page.
11. The user must be able to navigate to each of the pages via the UI. In other words, you must provide links to the add form, details page, update page, and a link to return to the index page.
12. You must add comments to describe the main parts of your PHP, HTML and CSS code.

Setup (Getting Started)

You have been provided with the skeleton code for the application on Blackboard: Assignment3-Skeleton.zip. This code provides you with the files for starting the project as well as the code for connecting to the database. You should download and extend this code rather starting from scratch.

You have also been provided with an SQL file to create the database and the table: assignment3.sql. You should use this file in phpMyAdmin to get the database running on your computer. Note that one example contact has been provided in the SQL file.

Setup Instructions

1. Download Assignment3-Skeleton.zip and unzip it.
2. Rename the extracted Assignment3-Skeleton folder to 320XXXX-Assignment3 (where 320XXXX is your student ID).
3. Move the 320XXXX-Assignment3 folder to your XAMPP htdocs folder.
4. Download the assignment3.sql file.
5. Open phpMyAdmin in your browser: http://localhost/phpmyadmin.
6. Navigate to the SQL tab.
7. Copy the entire content of assignment3.sql and paste it into the main query window.
8. Click the Go button in the bottom right of the query window.
9. Refresh phpMyAdmin by clicking the circular arrow underneath the phpMyAdmin logo. You should now see the compx222-assignment3-2024 database in the database list.
10. Open your browser and visit: http://localhost/320XXXX-Assignment3/index.php (replace 320XXXX with your student ID).
11. If everything has worked, you should see a message saying: "You are now ready to get started!"
12. Open the 320XXXX-Assignment3 folder in your preferred code editor and start extending the code to build your application.

When building your application, try to implement it one function at a time. Start by displaying all the contacts on the homepage. Once this is working, move on to displaying an individual contact on the details page, then adding a contact, then updating a contact, and finally deleting a contact. Make sure each function works before moving on to the next. If any errors occur, try to understand the cause, and fix them.

Restrictions

The purpose of this assignment is to demonstrate your ability to use PHP, MySQL, HTML and CSS. Therefore, the following restrictions apply:

- **DO NOT** copy and paste large blocks of PHP, SQL, HTML or CSS from other sources.
- **DO NOT** use visual design tools (such as Dreamweaver) that generate the HTML/CSS code.
- **DO NOT** use third-party frameworks or libraries (for PHP, HTML or CSS). This includes frameworks such as Laravel and Bootstrap.
- **DO NOT** use the HTML required attribute as this could prevent some of your PHP validation being triggered.
- **DO NOT** use any JavaScript for this assignment.

Guidelines

Project Organisation

Organize your project appropriately. In the root folder, include PHP files for your application's pages (add.php, details.php, index.php, update.php), a 'css' folder for CSS files, an 'img' folder for images, a 'config' folder for PHP code connecting to your database (db_connect.php), and a 'templates' folder for header and footer PHP files. The db_connect.php, header.php and footer.php files should be included in the PHP files for each of your pages. Most of this structure has been provided for you in the starter project code: Assignment3-Skeleton.

Your folder structure should look like this:

```
/root
   /config
      db_connection.php
   /css
      style.css
   /img
      image1.jpg
      …
   /templates
      footer.php
      header.php
   add.php
   details.php
   index.php
   update.php
```

Filenames (including images) should only contain ASCII characters, so you should avoid using Chinese characters. This is to prevent problems when zipping/unzipping.

Images

Do not use absolute filenames for image and videos because if you do, they will only work on your computer. All links to images and videos should use relative filenames.

Correct (relative):

```
src="img/image1.jpg"
```

Incorrect (absolute):

```
src="/Users/alex/Documents/your-website/img/image1.jpg"
src="C:\your-website\img\image1.jpg"
```

All **images should be appropriately sized for the web** and be no larger than they need to be. Large image files can significantly impact page load times, leading to a poor browsing experience for users.

CSS

You must use an external CSS file (style.css) for all of your CSS code then include it in your HTML file using a <link> tag inside the <head>. Do not include any CSS code in the HTML file (either inline in the HTML elements or in style tags at the top of the HTML page).

Correct:

```
<head>
    <link rel="stylesheet" href="css/style.css">
</head>
```

Incorrect:

```
<p style="color: green">

<style> ... </style>
```

HTML

ALL text must be enclosed in the correct HTML tags. Do not leave any un-tagged text in the HTML file. All HTML must be valid HTML5 and you should use semantic HTML5 elements where appropriate.

PHP

Use meaningful variable and function names. This will make it easier to read your code and keep track of what each part does.

Each page will contain some content that needs to be generated dynamically on the server via PHP. Each page should, therefore, be a .php file rather than an .html file. Most of the page will be HTML markup, but PHP code segments will occur at appropriate points and be placed between <?php and ?> tags.

PHP scripts can redirect the browser to other pages by setting the HTTP Location header using the header function. For example, header('Location: index.php'); redirects to the index.php page.

MySQL

You can test your queries in phpMyAdmin (using real values) to make sure they are correct.

All interactions with the database will be done via PHP. Make sure you use the correct PHP MySQLi functions to achieve this.

What to Submit

All your material for this assignment must be submitted using Blackboard.

1. Create a folder with a name of "320XXXX-Assignment3," where 320XXXX is your student ID. You should end up with a single ZIP file called **320XXXX-Assignment3.zip** containing your website files inside (see the Project Organisation section for details). Use only ASCII characters for filenames (no Chinese characters).
2. Submit the ZIP archive via Blackboard. On the course page you will find a section 'Assignments' with a link entitled 'Assignment 3 Submission' (or similar). Follow the link and upload your ZIP file.

No other mechanism for submission will be accepted.

NOTE: your site should work 'as is' when unzipped from your ZIP archive into the Apache htdocs folder, without any modification or changes to the database credentials. You should check that this is the case by unzipping your archive and testing the site yourself.

Assessment

Your mark for this assignment contributes to **30% of your overall grade**. Zip your website up into a single file and submit it on Blackboard before the due date. The marking will be as follows:

HTML Markup Including structure of the HTML document, appropriate form elements, attributes and input types	15%
CSS Styling Including overall style and consistency, usability, responsiveness, and use of selectors	15%
PHP Including validation, displaying or error messages, use of functions and appropriate techniques, absence of errors and warnings, all required pages included	40%
MySQL and PHP interaction Including support for each CRUD operation, properly structured queries, correct use of PHP MySQLi functions	20%
Project Structure Including file and folder organisation, file naming and external CSS	10%
Total	100%

You will lose marks if you break any of the restrictions or guidelines.

The assignment is **INDIVIDUAL work**. You are not allowed to work with other students or copy code from other sources and you must not share your code with any other student. If any assignment description is not clear to you, please ask your teachers.