INFT2100 – Assignment 2

Team Size: **2-3** Team Members Per Group (mandatory)
Project Name: INFT2100-<*Group Name*>

Objective:

To extend the capabilities of the Web Authentication Portal developed in Assignment , by integrating advanced database operations, third-party payment systems (PayPal), and image processing.

This assignment will give you a deeper understanding of database management, third-party PayPal integration, and handling complex data types like images in web applications, along with reinforcing best practices in web security and responsive design.

Brief Assignment Instructions and Requirements:

Below you will find the assignment requirements have been broken up for you to more easily digest.

1. Database Population and Management

Scripting for Database Population:

- Refactor your assignment 1 solution, to automatically populate the PostgreSQL database with realistic data. For example, author PHP code, that utilizes FakerPHP to generate a wide range of user data.

Database Management Techniques:

- Implement additional features in your administration dashboard, such as:
  - User data modification
  - User deletion
  - More advanced User querying (creativity welcomed).

2. Integration of Online Banking Features

Setting Up a PayPal Sandbox:

- Integrate PayPal's sandbox environment into your website for simulated financial transactions.
- Create a mock online ecommerce page/section in the dashboard where users can purchase items, backed using your PayPal sandbox account.
- Remember your PayPal account has both Business and Personal credentials, this will prove useful for demonstrations videos.

Transaction Handling:

- Write scripts to handle mock transactions and update user balances accordingly.

3. Handling Browser-Supported Images

Image Upload and Processing:

- Develop a functionality to allow users to upload an image in the following formats (exclusively JPEG, PNG).
  - A possible use case would be to allow users to upload profile pictures, but once again this is just a suggestion and creativity is welcome.
- Implement server-side image processing to the uploaded images using PHP.

Storing and Retrieving Images:

- Offer users two options for image storage:
  - As a Binary Large Object (BLOB) in the PostgreSQL database.
  - As a file path reference, storing the actual file in a server directory.

4. Things to Consider

Responsive Design:

- Ensure all new interface features are fully responsive and adhere to modern web design standards.

Security Considerations:

- Emphasize secure handling of data and images in your solution.
- Implement and demonstrate best practices in data privacy and security.

Solution Breakdown and Deliverables:

The following breakdown below, is provided to further assist you in understanding the intent and purpose of each section, by providing clear direction on what needs to be implemented within each.

Part 1: Database Population and Management

A. Scripting for Database Population

1. **Setup and Introduction to FakerPHP**
   - Install **FakerPHP** using Composer.
   - Create a new PHP file **db_population_script.php** for the script.
2. **Database Connection**
   - Utilize existing database connection logic from Assignment 1.
3. **Generating Mock Data**
   - For example, use FakerPHP to generate user data (names, emails, hashed passwords).
4. **Inserting Data into Database**
   - Write SQL INSERT statements in the script.
   - Execute the script to populate your table.

B. Database Management Techniques

1. **Enhancement of Dashboard**
   - Modify **dashboard.php** or create **admin_dashboard.php**.
2. **User Data Modification**
   - Implement a form for editing user details.
   - Process form submissions and update the database.
3. **User Data Deletion**
   - Add functionality to delete users from the database.
4. **Advanced Querying Interface**
   - Develop a page for custom user searches.

Part 2: Integration of Online Banking Features

A. PayPal Sandbox Integration

1. **Setup PayPal Sandbox**
   - Set up a PayPal sandbox account in anticipation of integrating it into your website.
2. **Transaction Handling**
   - Write PHP pages/scripts for handling mock transactions.

- You will want a form to take an order and mock a transaction

B. Mock Online Banking Section

1. **Dashboard Update** ✓
   - Add a section in the dashboard for your ecommerce (PayPal sandbox) activities.
   - A level of creativity is permitted here. I just want to see an ecommerce element added to your site, some action that invokes a PayPal transaction.

Part 3: Handling Browser-Supported Images

A. Image Upload and Processing

1. **Image Upload Feature** ✓
   - Implement an image upload form in the **user profile** section.

B. Storing and Retrieving Images

1. **Storage Options** ✓
   - Allow storing images as BLOBs **or** file path references.
2. **Retrieval and Display** ✓
   - Allow for the retrieval and displaying images on the website.

Submission:

1. A **private** git repository containing all your code, properly documented.
    - Please ensure to add me as a collaborator (**Sergio.Santilli@durhamcollege.ca**)
    - Paste the URL to the code repository within DCConnect as part of you submission

2. **Video Requirement**
    - Create a Short Video presentation. Your presentation should start with an introduction, where it must display a PowerPoint (or Google Presentation), that is 1 (single) slide. The slide introduces each member of your group, again, at the very start of your video.

    - The first (and only) slide of your presentation must include current images of you and your partner(s) (no avatars allowed) that are displayed appropriately. You must also include your Full Names, Student IDs, the Course Code, Course Name, Course Section and your Assignment information.

    - Within the recording, you or your partner(s) will take turns demonstrating your program's functionality. You must show your site working properly. You will also construct an assignment status report, a single page checklist/report. Use the report during the video, to facilitate communication confirming where requirements where successfully implemented and/or where requirements failed to be implemented and why.

    - Sound for your video must at an appropriate level so that your voices may be clearly heard, and your screen resolution should be set so that your program's code and console details are clearly visible. In short, QA your videos. If your video is poor, assignment failures can/will be assigned.

    - Your video should run no more than **~5-10** minutes. If you exceed this time, I simply will **NOT** be able to watch them... resulting in a grade of **zero**.

- **One** team member **per group**, must submit the following components to **DCConnect** on behalf of the entire team:
    1. The (zipped) project source code - **mandatory**
    2. The URL to the project Private Git Repository - **mandatory**
    3. The group video file - **mandatory**
        i. You may find OBS Studio useful to create this

Evaluation Criteria:

- Code Quality: Clear, readable, and maintainable code.
- Functionality: All described features should be implemented and working.
- Robustness: Proper error handling and usage of the Circuit Breaker pattern.
- Documentation: Both in-code comments and the accompanying report.

Good luck, and remember to enjoy the process!