UECS2094/UECS2194 WEB APPLICATION DEVELOPMENT

Exercise on Server-side Scripting

Task 1: Complete the *Project Updates* and *Traffic Announcements* Pages

This task is a continuation from Lab 03 Task 2. In this exercise, you required to complete the *Project Updates* and *Traffic Announcements* pages. This task also requires you to use MySQL database to store and retrieve data about project updates and traffic announcements.

1. Using a MySQL client (such as PhpMyAdmin), perform the following:
   a. Create a database named **uecs2094_pie**, the collation must be set to **utf8_general_ci**.
   b. Create a table named **announcement** in the **uecs2094_pie** database. The table structure is as shown in Table 1.1. You should write the SQL CREATE TABLE statement and save it in a file named **db.sql**.

Attribute	**Type**	**Key**
id	INT(11) AUTO_INCREMENT	PRIMARY
subject	VARCHAR(255)	
message	TEXT	
type	CHAR(1)	
posted	DATETIME	

Figure 1.1

NOTE: The **type** attribute is used to store the character **P** for Project Updates and **T** for Traffic Announcements.

2. Create a new PHP script named **admin/index.php** to implement CRUD (Create, Retrieve, Update, Delete) operations for the table named **announcement**. The **posted** attribute should take the current date and time without user input.

   Reference for PHP Date & Time:
   - http://php.net/manual/en/function.date.php
   - http://www.w3schools.com/php/php_date.asp

3. Complete the ***Project Updates*** and ***Traffic Announcements*** pages to display the posted date and subject of the updates/announcements. Clicking on the subject will display the update/announcement details, i.e. the subject, message and posted date.

Task 2: File Handling

1. Suppose that we need a simple web app that stores username, email address and gender of each user in a plain text file with the data for each user stored line by line while the attributes are tab-separated. Write a PHP script that reads the file and display a list of username, email address and gender in an HTML Table.

2. Based on the simple web app in Question 1, write a PHP script that inserts the record of a user into the text file. The input shall be obtained via an HTML Form.

3. Rewrite the PHP scripts in Question 1 and Question 2 where the text file format is now using Comma-Separated Values (CSV).

Task 3: Cookies & Session: User Authentication

1. Suppose that information about users in a web app is stored in a database in the table **user** that has the following structure:

   ```
   CREATE TABLE user (
      id          INT(11) NOT NULL AUTO_INCREMENT,
      email       VARCHAR(255) NOT NULL,
      password    VARCHAR(255) NOT NULL,
      PRIMARY KEY (id),
      UNIQUE KEY email (email)
   );
   ```

   The file **database.sql** provided to you contains the above **CREATE TABLE** statement and three **INSERT** statements to insert an initial three records to the table **user**.

   Please note that the *password* that we store in the table has been hashed using MD5 hash, in other words, the password that is actually stored is not the actual password but the MD5 hash value of the actual password. In PHP, we use the **md5()** function to generate the hash value of a string.

   Users are required to provide their *email address* and *password* to log in to the web app.

   a. Write a PHP script named **login.php** that accepts the *email address* and *password* via an HTML Form, then authenticate the user using data from the **user** table. Display an appropriate error message if the login credentials are incorrect. Otherwise, if the user provides the correct login credentials, redirect the user to the page **account.php** (You will write the script for this in part b).

The user authentication and authorization process for this PHP script should use PHP Session.

NOTE:
There are many techniques how we can keep track of the user using PHP Session. A simple method is to simply store the *id* of the user (from database) as session data. Every subsequent access to any PHP script that requires authorization will require code that reads the *id* value stored in the session variable earlier.

b. Write the **account.php** script. In this script, check that the user is already logged in. If the user is already logged in, simply display a welcome message and the *email address* of the user. Otherwise, redirect the user to **login.php.**

c. Write a PHP script **logout.php** to log the user out.

2. A more secured implementation of user authentication and authorization will require storing both the *id* of the user and identification, in this case, the *email address* as session data. Every subsequent access to any PHP script that requires authorization will require a query to the database to match both the *id* and *email address* before access is granted.

   Rewrite all the scripts in Question 1 to use this technique.

Task 4: Cookies & Session: Shopping Cart (Optional Exercise)

Please study through the tutorial on "Creating Simple Shopping Cart with PHP" from the web page http://www.sanwebe.com/2013/06/creating-simple-shopping-cart-with-php/comment-page-1

Try to follow the tutorial and attempt to build your own shopping cart web app.