# PHP and MySQL

## EXERCISE 5: PHP Sessions and authentication

Web servers receive requests from clients and send back a response. For example, the client might say “Can I have the web page whose name is *InterestingStuff.html*?” and the web server says “Sure, here it is,” and sends the HTML file InterestingStuff.html to them. Or the client might say “Can I have the web page whose name is *photoalbum-server-version-d.php*?” in which case the web server says “Sure,” then executes the PHP code in *photoalbum-server-version-d.php* which contacts the database and lists out the photographs and produces HTML output which the web server sends back to the client. However, web servers do not **remember** specific clients between requests.

There are many cases where it is useful or important for the web server to remember specific clients between requests. The most important is for controlling access. Suppose we want to control access to certain resources. What we could do is create a database table called ‘users’ which contains a list of usernames and passwords. Then we could give authorised people usernames and passwords and require them to enter them into a form on a web page. This is OK, but we then need to remember that they have entered their password already as they move around the site.

Here is a quick example:

Save the following as **poetry-form.html**:

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <title>Poetry Form</title>  </head>  <body>  <form action='poetry.php' method='post'>  First name <input type='text' name='firstname'>  Last name <input type='text' name='lastname'>  <input type='submit'>  </form>  </body>  </html> |

Save the following as **poetry.php**:

|  |
| --- |
| <html>  <head>  <title>Poetry</title>  </head>  <body>  <?php  print "<h1>A Poem for ".$\_POST['firstname']." ".$\_POST['lastname']."</h1>\n";  $nouns = array( "cat", "dog", "chimney", "floor", "bicycle", "cheese", "tree", "cloud", "sun", "moon", "balloon");  $verbs = array( "ran", "cried", "laughed", "loved", "ate", "stole", "bounced", "shone", "walked", "awoke");  $adj = array( "brown", "big", "small", "shy", "happy", "old", "young", "brave", "wise", "ancient", "orange", "kind");  $adv = array( "quickly", "slowly", "gracefully", "carefully", "enthusiastically", "foolishly", "forgetfully");  print "<p>";  for( $i = 0; $i < 4; $i++) {  print "The ".$adj[ rand(0, count($adj)- 1)]." ".$nouns[ rand(0, count($nouns)- 1)]." ".$adv[ rand(0, count($adv)- 1)]." ".$verbs[ rand(0, count($verbs)- 1)]."<br>\n";  }  print "</p>\n";  print "<p><a href='?'>Another</a></p>\n";  ?>  </body>  </html> |

Upload them both to the server. View **poetry-form.html**, type in your first and last names and press submit. The form calls **poetry.php** which prints your name in the title and then a random “poem”. The problem is that when you click the ‘Another’ link, the name you entered into the form is gone. If the form had been a username/password form to log in you would no longer be logged in.

Now change **poetry.php** to have this code at the start of the PHP section. Note that the line that printed out your name has been deleted and replaced by a new version that uses the $\_SESSION array instead of $\_POST.

|  |
| --- |
| session\_start();  if ( isset($\_POST['firstname'])) {  $\_SESSION['firstname'] = $\_POST['firstname'];  }  if ( isset($\_POST['lastname'])) {  $\_SESSION['lastname'] = $\_POST['lastname'];  }  print "<h1>A Poem for ".$\_SESSION['firstname']." ".$\_SESSION['lastname']."</h1>\n";  ~~print "<h1>A Poem for ".$\_POST['firstname']." ".$\_POST['lastname']."</h1>\n";~~ |

Go back to **poetry-form.html** and enter your name again. This time the script remembers your name when you click the ‘Another’ link. It has stored it on the server in the special $\_SESSION array.

Let’s make a proper log in script. To start with we need a database table to add users to. To create one easily save the following as **create-users-db.php**, upload it to the server and view it in a web browser.

|  |
| --- |
| <?php  require\_once("lib/dbutils.php");  $pdo = connect();  if ( !tableExists( $pdo, 'users')) {  print "<p>Creating users table...";  $sql = "CREATE TABLE IF NOT EXISTS `users` (  `user` varchar(255) NOT NULL,  `password` varchar(255) NOT NULL,  `role` enum( 'normal','admin') NOT NULL,  PRIMARY KEY (`user`)  )";  $pdo->exec( $sql);  print "<p>Inserting default admin user admin/password... ";    $sql = "INSERT INTO users SET `user`='admin', `password`='password', `role`='admin'";  $pdo->exec( $sql);    }  print "<p>Done";  ?> |

This creates a table called **users** and adds one user to it, whose username is *admin* and whose password is *password*. Check that it exists in PHPMyAdmin.

Now add the following two functions to **dbutils.php**:

|  |
| --- |
| function setUpSession() {  session\_start();  if ( !isset( $\_SESSION['user'])) {  $\_SESSION['user'] = "anonymous";  }  if ( !isset( $\_SESSION['role'])) {  $\_SESSION['role'] = "anonymous";  }  }  /\*  Session variables are set to anonymous IF THEY HAVE NO VALUE or if  a log in attempt is made and the credentials fail, or if a logout  attempt is made. If nothing is attempted and values are set, they  stay the same.  \*/  function logInOrOut( $pdo, $usersTable) {  setUpSession();    $action = "none";  if ( isset( $\_POST['formid']) && $\_POST['formid'] == 'login') {  $stmt = $pdo->prepare("SELECT \* FROM `$usersTable` WHERE user=? AND password=?");  $stmt->execute( array( $\_POST['username'], $\_POST['password']));  $rows = $stmt->fetchAll(PDO::FETCH\_ASSOC);  if ( count( $rows) == 1) {  $\_SESSION['user'] = $rows[0]['user'];  $\_SESSION['role'] = $rows[0]['role'];  $action = "login success";  } else if ( count( $rows) == 0) {  $action = "login failure";  $\_SESSION['user'] = "anonymous";  $\_SESSION['role'] = "anonymous";  } else {  die( "System error: ".count( $rows)." rows in database for user ".$\_POST['username']);  }  } else if ( isset( $\_REQUEST['logout'])) {  $\_SESSION['user'] = "anonymous";  $\_SESSION['role'] = "anonymous";  $action = "logout";  }  return $action;  } |

Once we have these functions available in **dbutils.php** and the users database table created, we can add session authentication to scripts quite easily. At the beginning of the script we can just put the following lines:

$pdo = connect();

logInOrOut( $pdo, 'users');

Then after that we can test whether the user is logged in by inspecting the $\_SESSION[‘user’] value. If it is ‘anonymous’ then no user is logged in. If not it will be the logged in user’s username. Here is a reworked version of the poetry script which uses this technique. If not logged in, it displays a log-in form, otherwise it displays a poem. The user stays logged in until they explicitly click the ‘Log out’ link. Save this as **login.php** and view it in a web browser.

|  |
| --- |
| <?php  require\_once("lib/dbutils.php");  ?>  <!DOCTYPE html>  <html>  <head>  <title>Log in demo</title>  </head>  <body>  <?php  $pdo = connect();  logInOrOut( $pdo, 'users');  if ( $\_SESSION['user'] == 'anonymous') {  ?>  <h1>Please log in to use this site</h1>  <form action='' method='post'>  Username <input type='text' name='username'>  Password <input type='password' name='password'>  <input type='hidden' name='formid' value='login'>  <input type='submit'>  </form>  <?php  } else {  print "<h1>You are logged in as ".$\_SESSION['user']."</h1>\n";    $nouns = array( "cat", "dog", "chimney", "floor", "bicycle", "cheese", "tree", "cloud", "sun", "moon", "balloon");  $verbs = array( "ran", "cried", "laughed", "loved", "ate", "stole", "bounced", "shone", "walked", "awoke");  $adj = array( "brown", "big", "small", "shy", "happy", "old", "young", "brave", "wise", "ancient", "orange", "kind");  $adv = array( "quickly", "slowly", "gracefully", "carefully", "enthusiastically", "foolishly", "forgetfully");  print "<p>";  for( $i = 0; $i < 4; $i++) {  print "The ".$adj[ rand(0, count($adj)- 1)]." ".$nouns[ rand(0, count($nouns)- 1)]." ".$adv[ rand(0, count($adv)- 1)]." ".$verbs[ rand(0, count($verbs)- 1)]."<br>\n";  }  print "</p>\n";  print "<p><a href='?'>Another</a></p>\n";    print "<p><a href='?logout=true'>Log out</a></p>\n";  }  ?>  </body>  </html> |

|  |
| --- |
| **EXERCISE 5: PHP Sessions and authentication** |
| ***Required in your report (basic exercise).*** |
| * A clickable link to poetry.php and login.php. * Reflection on the tasks and all activities involved. |
| ***Extended Tasks*** |
| * There are no specific extended tasks for this exercise. |
| *Original additional work:*  *Investigate and experiment with any related subject matter that interests you.* |