PHP Exercises – Secure areas of web sites, Sessions and cookies

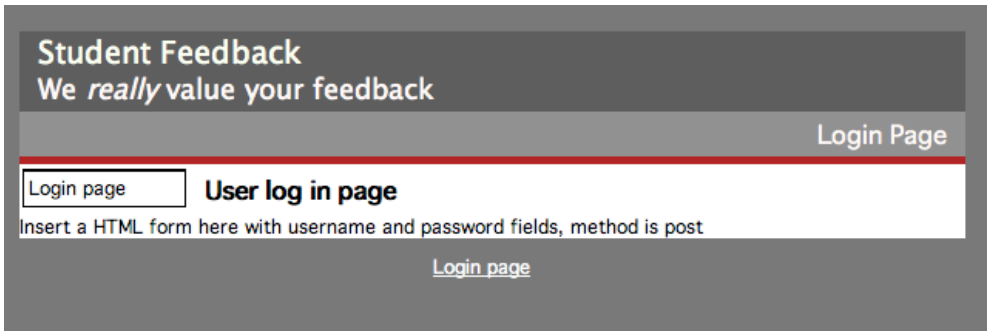
One
In your web publishing area, create a folder/directory called *lab9*. You will need to place all the files and folders used in this lab inside this folder so as to not conflict with previous lab work. Download the codeforlab.zip file from *barryavery.com.* Extract the files and place them in the following places:

- login.php INTO lab9
- main.php INTO lab9
- failed.php INTO lab9
- logout.php INTO lab9
- createUserdetails.php INTO lab9

Now create (inside lab9) a folder called *resources* and move the files as follows

- layout.php INTO resources
- database.php INTO resources
- styles.css INTO resources

Open *login.php* in a browser to see if everything is working. You should get a working *login* page that looks like this:



Two – Altering login.php to add username and password fields
Secure sites that require user authentication have a login form that generally requires (at least) a username and password.

Open *login.php* in an appropriate text editor.

In the space indicated, add a form with *username* and *password* fields -

```
<form name="form1" id="form1" method="post" action="loginresponse.php">

Username: <input name="username" type="text" id="username" />
Password: <input name="password" type="password" id="password" />

<input type="submit" name="Submit" value="Login" /></td>
```

Save the file and check it in a browser to see if it displays correctly.

Three – creating a database
Use the php web interface to create a database called *lab9*

Four – Getting database.php to work correctly
The next part will require the use of *database.php*. You will have to insert your username, database name and password into the correct line.

Open *database.php* (inside the *resources* folder) in an appropriate text editor and change the appropriate lines to have correct values for *username*, *password* and *databasename*.

Five – Testing the database connection and creating a *userdetails* table
Using a browser, view *createUserdetailsTable.php*

This will:

Create a connection to your database
Create a table called *userdetails*
Insert a sample user into *userdetails*
Close the connection

userdetails

userno	username	password
1	user1	qwerty

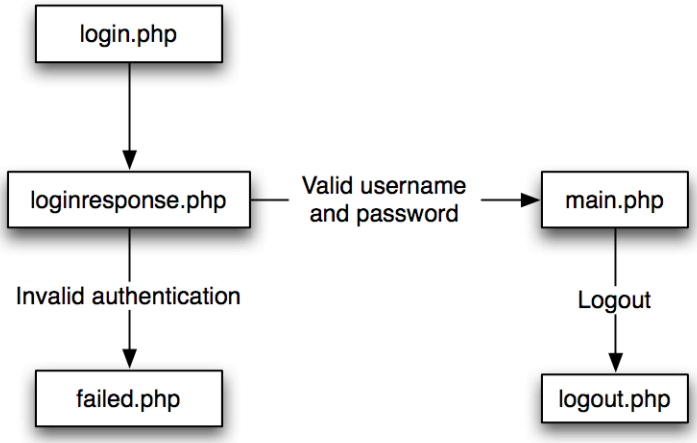
You can check that this table has been created correctly by using either using the mySQL monitor or by using the PHP web interface to mysql with the following SQL:

```
select * from userdetails
```

The *loginresponses.php* page
This structure uses *loginresponses.php* to test for correct authentication and then redirects to either the *main.php* or *failed.php* pages.

loginresponses.php will have this algorithm:

- If blank username or password redirect to failed
- If invalid username or password redirect to failed
- Start the session
- Redirect to main.php



Six – create *loginresponses.php* page
Use and appropriate text editor to create the loginresponses.php file. The first part of the file has to check for two entered values

- If blank username or password redirect to failed

Use these lines to do this:

```
if (($_POST['username']=="") or ($_POST['password']==""))
      {
      header('Location: failed.php');
      exit();
      }
```

Test this out by viewing *login.php*, then click on the button without entering a username or password.

Seven – test for correct username and password
The next part of *loginresponses.php* has to test for a valid username and password using the mySQL database. Place this code underneath the code in part five.

```
include "resources/database.php";

$result1 = queryDatabase("SELECT * FROM userdetails where username=
'".strtolower($_POST['username'])."';");

$row1= mysql_fetch_array($result1);
if ($_POST['password']!=$row1['password'])
      {
      header('Location: failed.php');
      exit();
      };
```

Test this out by viewing *login.php*, then click on the button entering an invalid username and password combination.

Eight – starting the session
The next part of *loginresponses.php* has to start the session and redirect to the main page. Place this code underneath the code in part six.

```
session_start();
$_SESSION['userno']=$row1['userno'];
$_SESSION['username']=strtolower($_POST['username']);
header('Location: main.php');
exit();
```

Test this out by viewing *login.php*, then click on the button entering a valid username and password combination – you should end up on *main.php*

Nine – securing main.php
Every page that you wish to make secure requires a test at the top of the page to test for the session (and accompanying cookie). Edit *main.php* to include these lines at the top:

```
session_start();
if ($_SESSION['username']=="")
      {
      header('Location: failed.php');
      exit();
      };
$userno=$_SESSION['userno'];
$username=$_SESSION['username'];
```

The last two lines are not really required, but demonstrate creating PHP variable from data held in the session file (in this instance the *userno* and *username*). These could then be used in the page somewhere.

Ten – logging out
Edit logout.php to delete the session and cookie file – place these lines at the top of the file:

```
session_start();
session_destroy();
```

Finally
Test out the complete set of pages, logging in and out using correct/incorrect authentication. Ensure that a logged out user cannot use the browser back button to gain access to the private areas of the site.