Worksheet 4
Query strings and Hidden Fields

The idea of this worksheet is to add download and order functionality to your HitTastic! site, making use of query strings and hidden fields.

Core Questions Part One : **Writing the Query String**

Use EPHP to test your work unless you are confident - it will help you identify issues with your code if it's not working correctly.

Adding a "download" link to your script

1. Load your ***searchresults.php*** script from Worksheets 2 and 3 Within the “while” loop which displays all the search results, add a hyperlink labelled “Download” which will link to the ***download.php*** script (not yet written), allowing the user to download that hit. For the moment, it should read something like:

```
echo "<a href='download.php'>Download this hit</a>";
```

Upload to Neptune and test.

Writing the Query String

2. Add a query string to your link so that it passes across the ID of the current song to a new script called ***download.php*** (which is not yet written). The query string variable name should be “songID” and the query string value should be the ID of the current song.

Optional fun exercise! (do ONLY if you finish the above within the <u>first 45 minutes</u> of the session)

Write a link to YouTube to show videos for each result. The link will be in this form:

```
https://www.youtube.com/results?search_query=<title>+<artist>
```

where <title> and <artist> are the title and artist from the database of the current hit (don’t actually type the angle brackets < and > !)

<u>Test the link to YouTube directly in the browser, not in EPHP - EPHP is not capable of loading external sites (i.e. YouTube).</u>

Core Questions Part Two : Writing the download.php script

You are now going to actually write the ***download.php*** script. Do not worry about correct use of GET and POST just yet, this is covered in the further questions, below. For the moment you are just going to use a GET request to run download.php.

3. The ***download.php*** script should actually download the user's chosen song. Downloading the song should increase the value of the "downloads" column in the database by 1.

The first thing you need to do is read in the information you passed across from the searchresults.php script. Look at the lecture notes on query strings to see how to obtain information sent across via a query string. Using this information, start writing the download script, so that it reads in this information into a variable called $id.

Next, increase the number of downloads of the song by one, represented by the ***downloads*** column in the database. You will need an SQL UPDATE statement for this. Add an appropriate UPDATE statement to your script.

THE SQL UPDATE should be as below (you need to complete it):

```
UPDATE ???? SET downloads=downloads+1 WHERE ????
```

b) Finally, complete your download script and test it.

Further questions

These questions allow you to use hidden fields and also allow you to re-write your purchasing system so that you use GET and POST appropriately.

5. This question allows you to practice with hidden form fields. Imagine that a user can order physical copies of the song (e.g. on vinyl or CD) as well as download it. A **quantity** column has been added to the wadsongs table to indicate the number of physical copies currently in stock.

Make a script called ***order1.php*** containing a form which allows the user to specify how many copies to order. Add a link to ***order1.php*** from searchresults.php. You should pass across the song ID from searchresults.php to ***order1.php*** using a query string, in the same way that you pass across the ID to download.php.

The ***order1.php*** script should read in the ID from the query string and echo out a form with two fields:
- a quantity field,
- and a hidden form field containing the ID.

The form should send its information to a new script, ***order2.php***, using a method of "**post**" (why?)

c) Write a script called ***order2.php***. It should read in the ID and quantity from the form in Question 4, and uses it to reduce the quantity in stock of physical copies of the selected hit by the selected amount.

6. Alter the ***order1.php*** script so that it displays the full details of the hit that the user has chosen. Use an SQL statement (what type?) to do this.

7. Using an **if** statement, add error checking to the ***order2.php*** script, so that the hit is not bought if the quantity available in stock is less than 0, and instead an error message appears.

8. **Functions *(this is an extension of an advanced exercise from session 2)***: If you have not done so already in earlier sessions, create a CSS layout containing a sidebar (can be a top bar if you prefer) and main content area.

Then, create a PHP *function* to write out the sidebar as HTML. This should take two parameters: the page title (e.g. "Main page", "Signup" etc), and an associative array of links (to other pages), where the key is the link text and the value, the page you're linking to. It should then write out a sidebar <div>, using the title and the associative array of links supplied to it.

Using this function, give each HTML and PHP script you've written so far (including the search results, the download script and the order form and script) a sidebar with a title and links. Ensure the title is an appropriate title for that page, and there are links to the search page, the signup page, and the "about" page (see worksheet 2).