# Advanced Higher Computing Science

## Web design and development: website design and PHP in-built functions workshop materials



This edition: December 2024 (version 1.0)

# Introduction

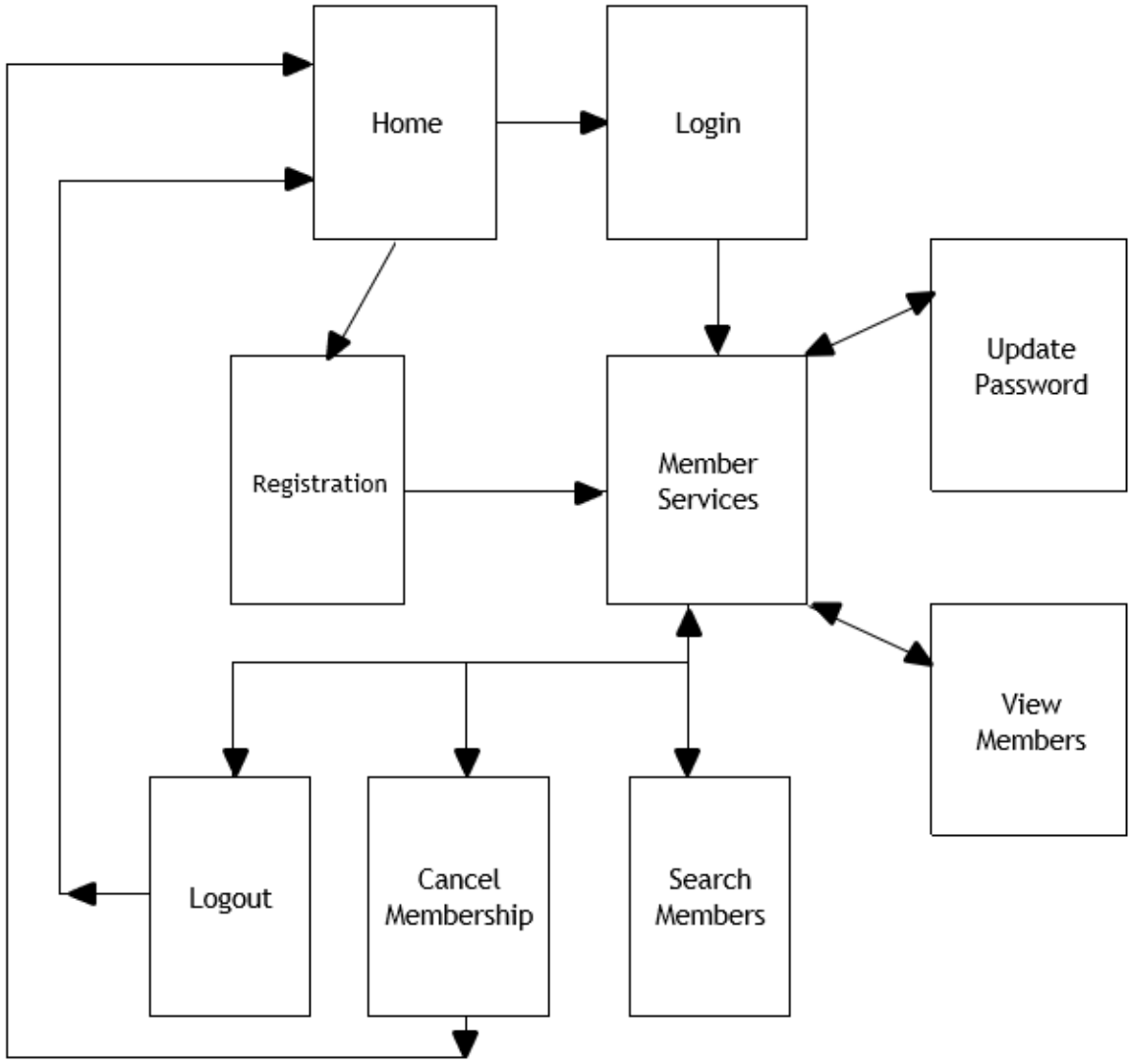
This document is for teachers and lecturers and/or Advanced Higher Computing Science candidates.

This document contains website design and PHP in-built functions questions and accompanying marking instructions devised for the workshop at an Understanding Standards event held in 2023.

# Questions

The Kookie Klub is a website dedicated to biscuits and cookies. Members can sign up to view details of other members and search for other members who share a love for the same favourite cookie.

The navigation structure of the website is shown below.



Once registered or logged into the site, members will see a short welcome message on each of the Member Services pages. This message will be personalised with the stored username.

An example of this is shown below:

AdaL, Welcome to the Member Search Page!

The registration and login pages use HTML form to allow website users to enter their details.

- on the page `registration.html`, users enter their username, email address, home town and password. They then select their favourite biscuit from a drop-down list of biscuits and cookies.
- on the page `login.html`, users enter their username and password.

When submitted, these forms POST the member's details to the pages `registration.php` and `login.php`.

**1.** Annotate the site navigation structure to show the passing of member details across the Kookie Klub website. Use the words POST and SESSION to indicate how the details are shared across the pages.

**2.** The design for the PHP page that will process the registration details is provided below:

1. start session
2. assign submitted values to variables $username, $email, $town, $password
3. assign $username to session variable
4. assign connection details to variables $server, $database, $user, $pass
5. create SQL query used to add user details to Member table and assign to variable $query
6. connect to DB server
7. execute SQL query and assign result to variable $insert
8. close DB connection
9. if query executes successfully then
10. display 'successful registration' message
11. else
12. display 'registration unsuccessful' message
13. display link to return to Registration page
14. end if
15. display link to 'Member Services' page

(a) Which in-built PHP variables would be needed to implement this design? State where each variable will be required.

Which in-built PHP functions would be used to implement:

(b) Line 1 of the design.

(c) Lines 6 and 8 of the design.

(d) Line 7 of the design.

(e) Describe how the design could be altered to include a check to ensure that the submitted username hasn't already been stored by another user.

**3.** (a) Create the design for the page `logout.php` that will be used to process and request to log out of the site.

(b) List the in-built variables and functions that will be needed to implement this page. You should indicate the line of the design where each function will be required.

**4.** (a) Create the design for the page `cancel.php` that is used to process requests to cancel a membership. Before cancelling their membership, members will be asked whether they wish to continue.

(b) List the in-built functions that will be needed to implement this page. You should indicate the line of the design where each function will be required.

**5.** Look at the design of the PHP page used to process the request to search for members.

1. start session
2. assign connection details to the variables $server, $database, $user, $pass
3. display personalised welcome message
4. display option to search for members who love the same type of biscuit
5. create SQL query to search for favourite biscuit for current user
6. connect to database
7. execute query and assign result to variable $favourite
8. create SQL query to search for username and town of members who love the same type of biscuit
9. execute SQL query and assign result to variable $results
10. close DB connection
11. assign number of results found to variable $howmany
12. if $howmany = 0 then
13. display 'No matching members found' message
14. else
15. display table headings to show results
16. while there are still query results to display
17. display member username and town
18. end loop
19. end if
20. display link to return to 'Member Services' page

Explain how the following in-built PHP functions would be used in the implementation of this page:

(a) `session_start()`

(b) `mysqli_connect()`, `die()` and `mysqli_close()`

(c) `mysqli_query()`

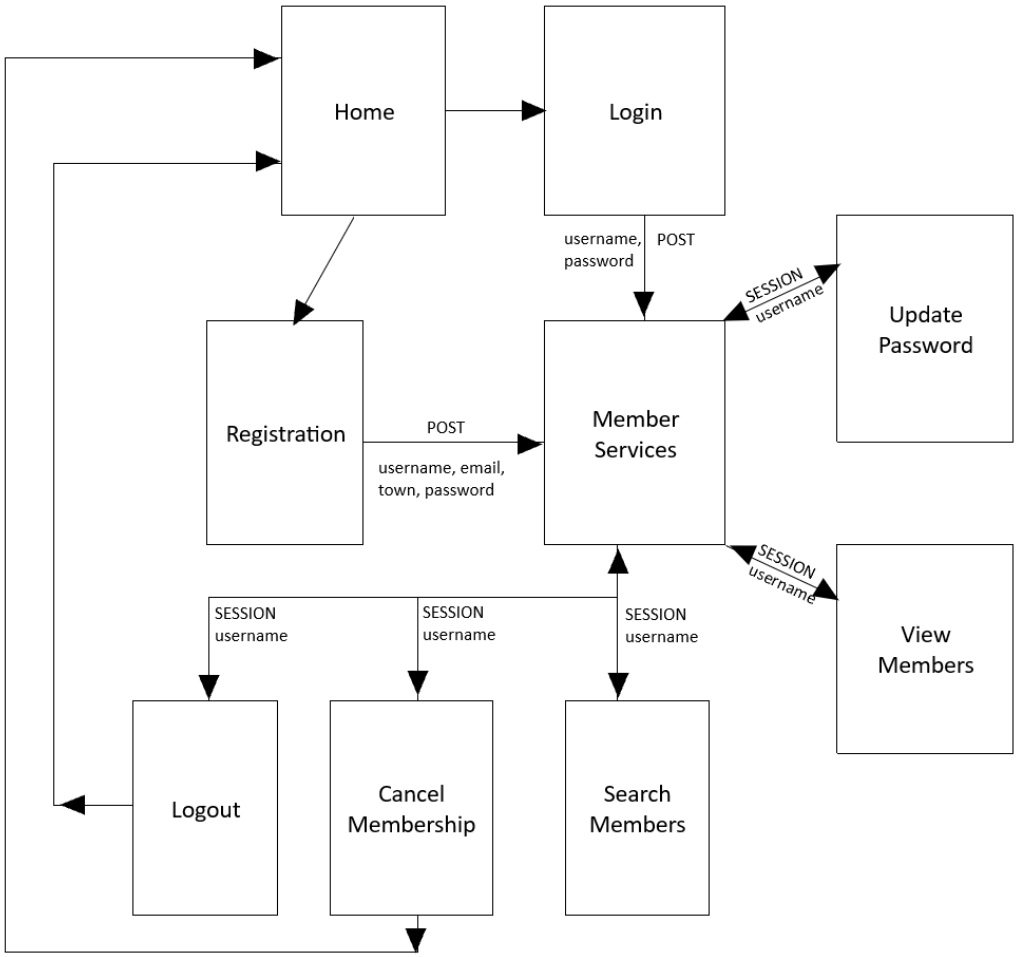
(d) `mysqli_num_rows()`

(e) `mysqli_fetch_array()`

(f) Explain why this page needs a session variable.

# Marking instructions

**1.**



**2.** (a) `$_POST` and `$_SESSION` variables will be needed.

`$_POST` will be needed at Line 2 and `$_SESSION` will be needed at Line 3.

(b) Line 1: `session_start()` function

(c) Line 6: `mysqli_connect()` and `die()` functions
Line 8: `mysqli_close()` function

(d) Line 7: `mysqli_query()` function and possibly `die()` function

(e) An additional query would be needed to select all records where stored username matches submitted username. This query would be executed before Line 4 of the design. If the query failed, then the submitted username must be unique and so the submitted details can be added to the Member table. However, if the query is successful, then the username has already been stored and the user should be asked to try a different username and returned to the Registration page to re-register.

**3.** (a)

1. start session
2. display personalised welcome message
3. display message to check whether user wants to continue with the log out request
4. if request is confirmed then
5. end session
6. display link to return to Home page
7. else
8. display link to return to 'Member Services' page
9. end if

(b) The global `$_SESSION` variable will be needed at Line 2.

At Line 1, the `session_start()` function is needed.

At Line 5, the `session_destroy()` function is needed.

**4.** (a)

1. start session
2. display personalised welcome message
3. assign connection details to variables $server, $database, $user, $pass
4. display message to check whether user wants to continue with the cancellation request
5. if request is confirmed then
6. create SQL query used to remove record from Member table and assign to variable $query
7. connect to DB server
8. execute SQL query and assign result to variable $delete
9. close DB connection
10. if query executes successfully then
11. display 'successful cancellation' message
12. end if
13. end session
14. display link to return to Home page
15. else
16. display link to return to 'Member Services' page
17. end if

(b) Line 1: `session_start()`

Line 7: `mysqli_connect()` and `die()`

Line 8: `mysqli_query()` and `die()`

Line 9: `mysqli_close()`

Line 13: `session_destroy()`

**5.** (a) This function is needed on this page to ensure that the global session array will be shared with the page so that a personalised message can be displayed.

(b) At Line 6, the `mysqli_connect()` function is used to connect to the database server; the `die()` function is used to terminate the execution of the current script and display an optional error message. At Line 10, `mysqli-close()` function is used to terminate the connection with the DB server which makes it available for other users.

(c) The `mysqli_query()` function is used at Lines 7 and 9 to execute the SQL queries and return results to the specified variables.

(d) The query that uses the favourite biscuit of the current member to search for matches may return zero, one or many records. The `mysqli_num_rows()` function is used at Line 11 to assign the number of rows in the query answer table to the specified variable. At Line 12, an `IF` statement is used to decide whether to display 'no matches' message or a table showing the results returned from the query.

(e) At Line 16, the `mysqli_fetch_array()` function is used to process one single row of the query answer table at a time. In this design, it is used to control the conditional loop that is used to display the query results.

(f) The username submitted is only available on the page it is submitted to – on any subsequent pages, the `$_POST` variables will be empty. However, a session variable that is assigned on the PHP page used to process the registration or login details can be shared across other pages of the site for the duration of the session. In this way, use of the global session variable means that the submitted `username` is available to the 'Member Search' page and can be used to generate a personalised welcome message.