

# 0351709- Smart\_And\_Connected\_Devices \_1 *by* Neil O`mahoney

---

|                |                                                                                                |                 |      |
|----------------|------------------------------------------------------------------------------------------------|-----------------|------|
| FILE           | 12740_NEIL_OMAHONEY_0351709-<br>SMART_AND_CONNECTED_DEVICES_1_236131_206102684.DOCX<br>(8.37M) |                 |      |
| TIME SUBMITTED | 27-FEB-2020 11:03PM (UTC+0000)                                                                 | WORD COUNT      | 1770 |
| SUBMISSION ID  | 120848325                                                                                      | CHARACTER COUNT | 8254 |

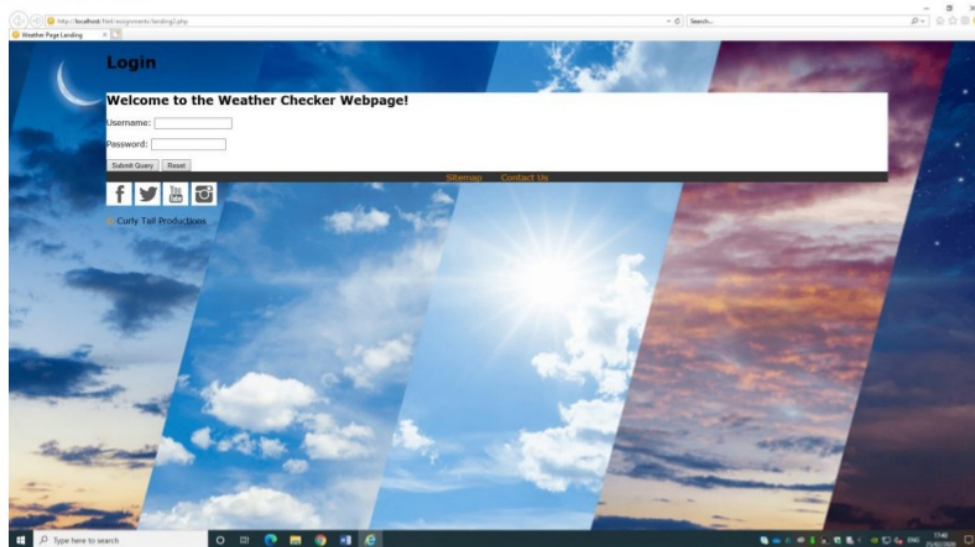
# Smart and Connected Devices 1

Neil O'Mahoney 0351709  
Cyber Security Part-Time Student

## Table of Contents

|                           |    |
|---------------------------|----|
| Landing Page .....        | 3  |
| Check login.....          | 4  |
| Conn Info.....            | 5  |
| Trim quotes function..... | 6  |
| Secret page .....         | 7  |
| Search page.....          | 8  |
| Transfer of Data .....    | 10 |
| Close session .....       | 11 |
| Conclusion.....           | 11 |
| References .....          | 12 |

## Landing Page



Above is the landing page. It features a username and password logon box.

The webpage has been designed in Cascading Style Sheets (CSS) and is HTML 5 compliant. Ideally the author should have used bootstrap to configure this element of the website. CSS was used as it supports multiple browsers and environments 'CSS supports various profiles in consideration of different device environments.' [1]

The username and logon box are linked to a MySQL database table called 'users'.

Initially at the top of the page a PHP script checks that the user isn't already logged in. The '<?' brackets are there to allow PHP to be typed in a HTML document. The logic of the statement reads that if the 'uname' is 'NOT EQUAL TO' (!=) "" then the user must be logged in and the session is still live, the ELSE statement is there for the possibility that the user might not be logged in and posts the form below. The IF ELSE statement is useful in that it can be used until a statement matches the condition '...You provide a sequence of conditions, the program can check each until it finds one that is true.' [2]

```
<div>
<?if (@$_SESSION["uname"]!="") //session variable which remains untill wiped.
{
echo "Welcome ".$_SESSION["uname"]."</h1>";
echo "<a href='logout2.php'>Log out</a>";
}
else //not logged in

{?>
```

The format of the questions presented to the user is put into a form. This is a standardised format which presents to the user where to input their information.

There is a submit button which pushes what data has been inputted into it to a page called checklogin2.php.

```
<form action = "checklogin2.php" method="post">

Username: <input type="text" name="username" autocomplete="off"> <br></br>

Password: <input type="password" name="password" autocomplete="off"> <br></br>

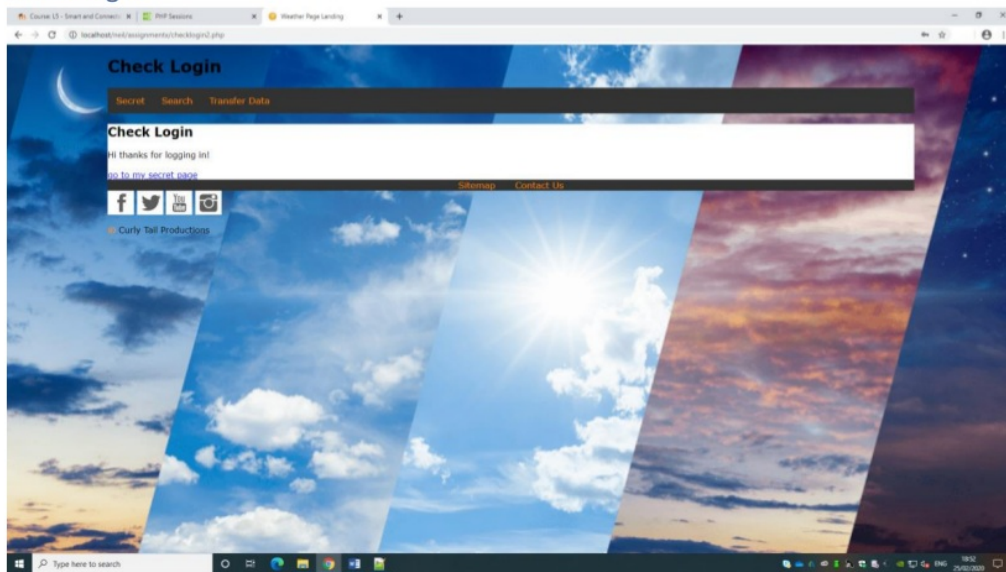
<input type = "submit">
<input type = "reset">

</form>
<?>
```

The password field is hidden by changing a field type to “password” in the HTML form. Autocomplete has also been switched off in the same way.

The method of sending the data is the ‘POST’ method. According to W3 schools this more secure than the Get method as the ‘...Parameters are not stored in browser history or in web server logs’. [3]

### Check login



SQL statement checks that the user is a valid member of the website against the database

```

$query="SELECT * FROM users WHERE username like '$un' AND password like '$pw'";

//echo$query;

$result=mysqli_query($link, $query);

$num=mysqli_num_rows($result);//how many rows found?

//echo $num;

//echo $un.$pw;

if($num==1)//login person
{
    $_SESSION["uname"]=$un;
    echo "Hi thanks for logging in!<br> <br>";?>
    <a href="secret2.php"> go to my secret page</a>
<?
}

else//dont log them in array

```

UN & PW are variables that the user has to type in. the 'IF' statement is a logical check of the above SQL statement that adds the user to a session if the account is found, containing the username of the person logged in. IF statements '..Condition is true, the following block of code will be executed.' [2]

```

else//dont log them in_array
{
    echo "sorry, wrong details";?>
    <br>
    <a href="landing2.php">try again</a>
<?
}
?>

```

Image above depicts the unsuccessful login of the user and demonstrates validation on the website.

### Conn Info

```

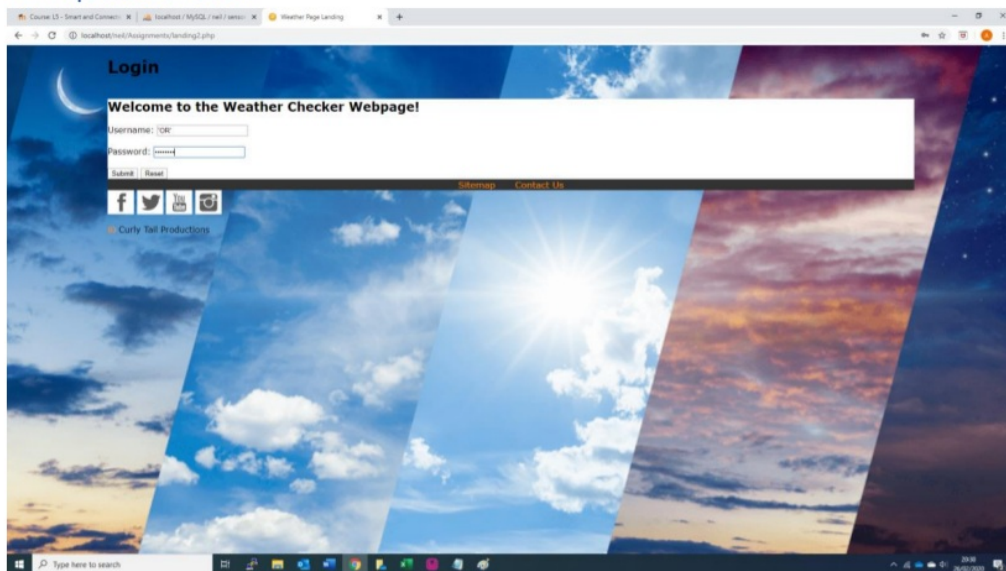
<?php
$strServer="localhost";
$strDatabase="neil"; // CHANGE TO YOUR DATABASE NAME HERE
$strUser="root";
$strPwd=""; // Leave blank for WAMPServer

$link=mysqli_connect($strServer,$strUser,$strPwd,$strDatabase)or die("Could not open database");
?>

```

The 'Conn info' file is referred to throughout the website to access the SQL database. This is a single file and any information that is changed here will change for all access pages. 'MySql\_connect' opens a new connection with the server, database, user and password credentials.

## Trim quotes function

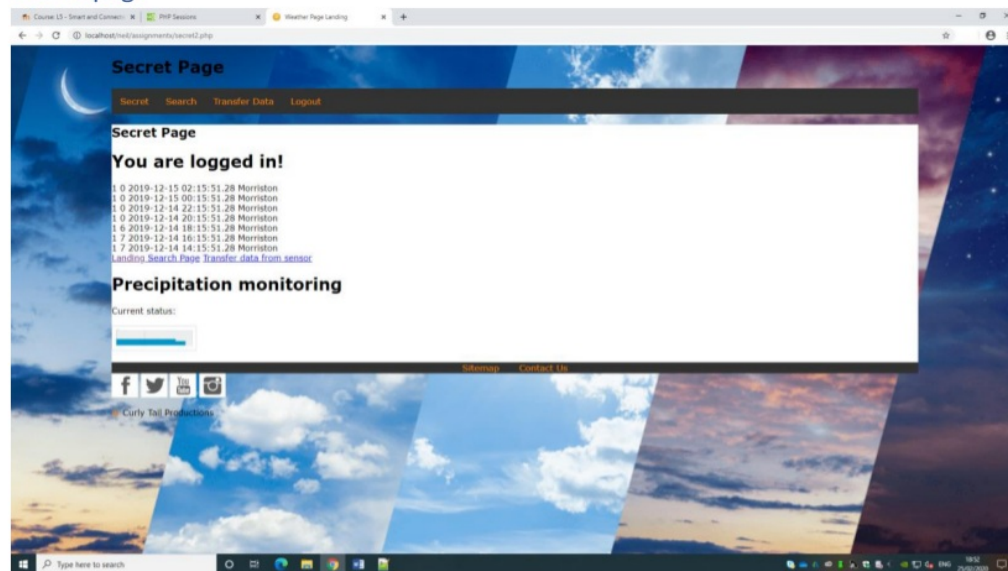


The above shows the user intentionally logging in with the 'OR' quotes function. This should cause the database to crash and allows the insertion of code. The password typed in is a correct one and had the trim quotes function not been used, would have allowed unrestricted access to the site.

```
$un=str_replace("'", " ", $un);  
$pw=str_replace("'", " ", $pw);
```

Above shows the trim quotes against the username and password. This statement unquotes a quoted string, the left in the brackets is what is quoted, the right is what it becomes.

## Secret page



The above shows the now logged in webpage, known as the secret page. So called because the only way to access this webpage is via a login. The data on the page is pulled from the database and displays information back to the client. Ideally this would be removed, however for testing purposes and to demonstrate that it worked, the author thought it prudent to display his database.

```
$query="SELECT * from rainfalldata,sensor WHERE rainfalldata.sensor=sensor.id"; //query which joins tables in rainfalldata and
sensor together to name the area where they are from not just a name
$result=mysqli_query($link,$query); //fetches back a row from an array
```

```
while($r=mysqli_fetch_array ($result))
{
    echo $r["id"]. " ";
    echo $r["mydata"]. " ";
    echo $r["thedata"]. " ";
    echo $r["sensorname"]. " <br>"; //joined table data
}

?>
```

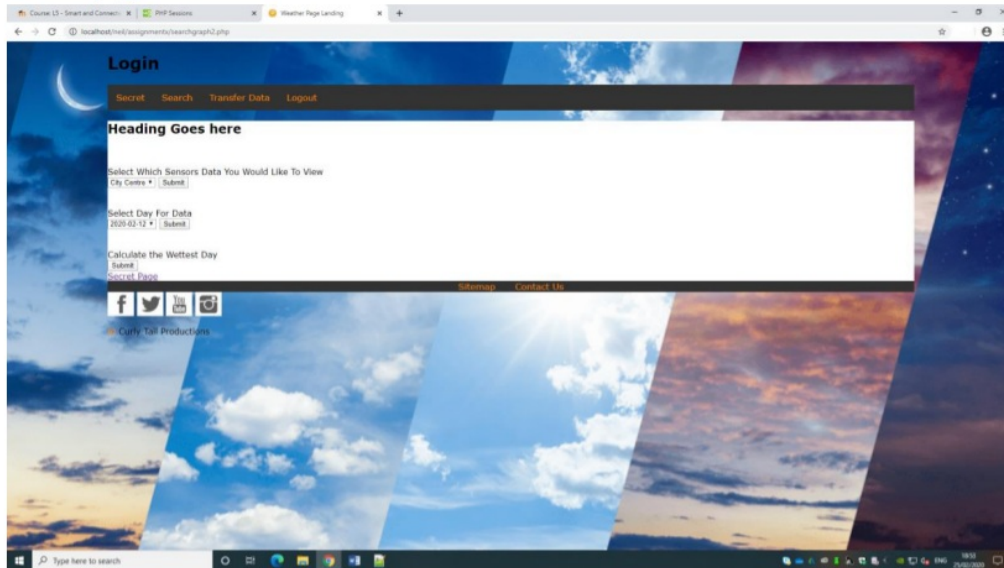
Shown on the page is a join between two SQL databases the sensor name, rainfall, timestamp and the location are show. The location is joined from the sensor table to data pulled from the text files. As we can see, the author has stated the two separate tables in the first part of the query then references the column names 'sensor' and 'id' later on. There needs to be data common to both (in the case of our table the sensor from the rainfalldata table and the id number from the sensor table [4] '...We can use this column to tie the two tables together into a single query.'

```
<?
echo "<h1>Precipitation monitoring</h1>";
echo "Current status:<br><br>";
?>
 |

<?
}
else
```

The author displayed the graph in a small pictorial format to show that there had been data connecting and creating a graph, it is referenced as an image file, pulled from 'makegraph.php'. The author should have changed the size on the graph at the time to display its full size 640 x 640 by using the width="640" height="640". The graph can be further shown on the 'Search' page.

## Search page



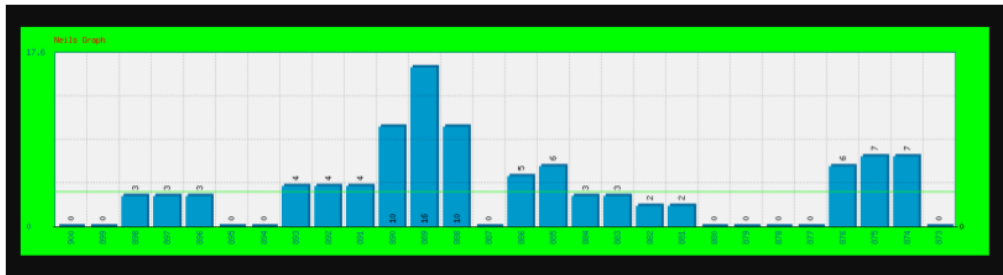
Containing multiple SQL queries, the 'Search Page' allows the user to change variables to manipulate the data that is found within the webpage.

```
Select Which Sensors Data You Would Like To View
<form action="makegraph7.php" method="get">
<select name="userinput">
<option value="2">City Centre</option>
<option value="1">Morrison</option>
</select>
<input type="submit">
</form>
```

The first query allows the user to search the different sensors, 'Morrison', 'City Centre' and 'Sketty'. These all relate to 1,2 and 3 'rainfalldata' text files. When the user selects the drop down and clicks search button, this relates to a number of the sensor and using the 'Get' command sends this number (pre-programmed into the HTML) to some SQL in 'makegraph7.php' seen below as the array variable '\$ui':

```
$query="SELECT id,mydata,theDate
FROM rainfalldata WHERE sensor like '$ui%'"; //like statement used and % so that bringing part of data is found.
```

In retrospect it would have been a better idea to populate the dropdown list from the rainfalldata.txt file rather than having it hard coded into the HTML as it might need to be changed to expand the search.



Displaying the data above is for Morriston or Datagraph #2. This is shown on a separate webpage. To display onto the webpage there is a 'while loop' which displays the data. Ideally it would display the date timestamp, however the author selected the ID which doesn't help the reader to discover when the timings were taken. It would be simple enough to fix/ Code below demonstrates this While Loop:

```
while ($r=mysqli_fetch_array($result)) //returns array to the fetched row and moves the pointer ahead.
{
    $data[$r["id"]]= $r["mydata"];          //put data into array
}
```

The graph displays a line of average/ medium, this was added to give the user an idea if it was above or below here it should be, added as an extra feature to the PHPmygraph page:

```
$points['average'] = $this->calculateAverage($dataStructure, $points);
```

To demonstrate the ability to search for a particular day, the author also added this function to his site, it mostly featured the same credentials as above, although the SQL statement searched on the date instead, the '%' is a wildcard which searches the first part and the second part of the timestamp doesn't have to be exact:

```
$query="SELECT id,mydata
FROM rainfalldata WHERE thedate like '$u1%'"; //like statement used and % so that being part of data is found.
```

Select Day For Data

```
<form action="makegraph5.php" method="get">
<select name="userinput">
<option value="2020-02-12">2020-02-12</option>
<option value="2019-12-15">2019-12-15</option>
</select>
<input type="submit">
</form>
```

The 'Get' method of moving data was used here. This is to show the data that was being passed in the URL to make it easier for the author to see what was happening.

The final search the author has created relates to the calculation of the wettest day. There is not information transferred from the webpage as is done previously it just runs the query from the linked webpage itself:

Calculate the Wettest Day

Submit

```
<br></br>
Calculate the Wettest Day
<form action="wettestday.php" method="get">
<input type="submit">
</form>
```

The query inside the SQL statement first calculates the total sum of mydata field across all in rainfalldata, they are grouped according to the datetimestamp into a decending order by using the 'ORDER BY' command. The author then displays the top result and displays according to what has been produced inside a while loop. 'ORDER BY' is desgined to rank by a specific column where as GROUP BY is desgined to summise combinations of columns.

The 'Order By' 'Sort returned results by one or more cols in ascending or decsending order'. [5]

```
$query="SELECT SUM(mydata), mydata, thedate, sensor
FROM rainfalldata
ORDER BY thedate DESC
LIMIT 1"; // adds all the days with a precepitation value from rainfalldata, sorts them
into order by their date, descending order then sets a limit of how many are displayed.

while($r=mysqli_fetch_array ($result)) //based on the above ina while loop displaying the gotten data.
{
    echo    "the wettest day had:<br>";
    echo $r["SUM(mydata)"]. "mm of rain!<br><br> ";
    echo    "and was on: <br>";
    echo $r["thedata"]. "<br> ";
}

?>
```

```
the wettest day had:
176mm of rain!
```

```
and was on:
2019-12-15 02:15:51.28
```

An improvement on this could have been to remove elements of the timestamp, but for reasons of time the author was unable to do so.

### Transfer of Data

There are 3 different files that need to be read into the database, the author created a loop for this with the element 'j' which can be substituted for the last number in the filename (as long as it matches rainfall). This is set from 1-3 in the 'For Loop':

```
for ($j=1; $j<=3; $j++) //goes from 1 to 3 to open files 1 - 3 in rainfall.
```

The file is opened, checked for contents inside (if not information displayed that it isn't) if it is the data is split into an array and read into the database. The variable of the rainfall number splits it into different fields for the sensor. Finally, the file is closed then the contents wiped so that it cannot be read again duplicating data, seen below:

```

if(filesize($myfile)>0) //checks file size against 0
{
    $data=fread($filelink,filesize($myfile)); //read entire file

    fclose($filelink); //close the file

    $incoming=array(); //declare array
    $incoming=explode(chr(13),$data); //splits data into an array

    $length=sizeof($incoming); //how big is the array?

    $tempdata=array(); //declare temp array

    for($x=0;$x<$length;$x++) //loops around for as many times as there is from incoming
    {
        $incoming[$x]=trim($incoming[$x]); //clean off the linebreak whitespace
        $tempdata=explode(" ",$incoming[$x]); //split current line of data that has been read into 3
        $timeofreading=$tempdata[1]. " " . $tempdata[2]; //stick datestamp back together in one variable ($timeofreading)
        $rainfall=trim($tempdata[0]); //rainfall reading
        $query="INSERT INTO rainfalldata(mydata,thedata,sensor)
        VALUES($rainfall,'$timeofreading',$x)"; //writes the value of the sensor according to the file it is opened from
        $result=mysqli_query($link,$query);
    }
    echo "Data has been transferred from rainfall $j<br>";
    $filelink=fopen($myfile,'w'); //effectively deletes all of the file contents
    fclose($filelink); //closes file.
}

```

## Close session

Here the session name is removed within the PHP, effectively blanking it. This is a security element so that the next person cannot hit the back button and use the account to change data.

```

<?session_start(); //accessing existing session
$_SESSION["uname"]=""; //set to empty

session_destroy();

header("location:landing2.php");//redirect automatically

?>

//Goodbye!<br><br>

//<a href="landing2.php">login page </a>

```

## Conclusion

The overall feel of the site is a little clunky, the HTML doesn't fit together properly and the author should have chosen an easier bootstrap program, due to time limitations this was not an option, however would have made editing it far easier later on.

The setting of the graph on the secret page should have been clearer and the width and height tags been applied to suit the size.

When drawing information to fill the drop-down lists, this should be done dynamically and be pulled from the SQL database. Hard coding into the HTML is not a good idea, as the dates could change, making more work for an administrator.

On the point of the graphs displaying the 'id' tag and not the 'timestamp' this was an easy change to make which should have been done to help the reader identify what they are looking at.

The calculations of the wettest day is functional, again too much information on the timestamp is a let-down, but using the 'ORDER BY' command which is compatible with this version of SQL as opposed to 'GROUP BY'.

The 'Conn Info' file is a good central location to store database access details rather than referring in and out of the php in the page making separate connections this is one central location to do so.

## References

- [1] H.-J. ., W.-Y. S.-M. S. Soon Bum Lim, "CSS3 extensions for setting web content in a 3D view volume and its stereoscopic 3D display," *Computer Standards and Interfaces*, vol. 50, pp. 65-75, 2017.
- [2] L. T. Luke Welling, *PHP and MYSQL Web Development*, USA: Pearson Education, 2009.
- [3] Unknown, "HTTP Request Methods," W3 Schools, 25 Feb 2020. [Online]. Available: [https://www.w3schools.com/tags/ref\\_httpmethods.asp](https://www.w3schools.com/tags/ref_httpmethods.asp). [Accessed 25 Feb 2020].
- [4] R. Nixon, *PHP, MySQL & Javascript*, CA: O'Rilley, 2009, p. 192.
- [5] R. Nixon, *PHP, MySql & Javascript*, CA: O'Reilly, 2009.

### FINAL GRADE

80/100

### GENERAL COMMENTS

#### Instructor

Data management - Good database structure with 3 tables and data transferred efficiently and deleted afterwards. Well done 10/10

Range of HTML/PHP coding demonstrated - Good overall with sensor and date searching. You have shown the wettest day across all sensors which is good, you could also limit this to individual sensors (as they could be in very different geographical locations) Login is working well. Could do with some headings on your data presentation

20/25

Visualisation of data using graphs - Good to see that you have used searches with your graphs, you could also embed your graphs into the page, (this is trickier and you'd need to use querystring and a get for future reference) Axes need to have date placed on them as it's difficult to understand the graph otherwise. 11/15

Overall integration of website - Generally good, although some pages are unstyled. You could also use includes for a more efficient page layout. Not sure it's appropriate to call the page "secret" for this application, just call it "dashboard" or something, I only called it secret in the example to make it clear it was unreachable without logging in.

7 /10

Knowledge and understanding overall - Good overall, just have a think about the points made above 16/20

Report - Quite clear with code extracts well presented to support your explanations. Don't be tempted to use "the author" to avoid using "I", you need the style to be passive, not active in academic reports. "The author has stated the two separate tables" -> "The two separate tables have been stated" Some good reflection and points for future work. 16/20

Overall - 80%

---

PAGE 1

---

PAGE 2

---

PAGE 3

---

PAGE 4

---

PAGE 5

---

PAGE 6

---

PAGE 7

---

PAGE 8

---

PAGE 9

---

PAGE 10

---

PAGE 11

---

PAGE 12

---