**PHP Basic Database Connection**

**Connecting to a MySQL database**

$conn = new mysqli($servername, $username, $password, $dbname);

The above connects to MySQL database using four parameters and stores the connection in the variable (object) $conn.

**Validate database connection**

if ($conn->connect\_error) {  
    die("Connection failed: " . $conn->connect\_error);  
}

The above code through an exception (message), if the connection can not be completed.

**Executive SQL statement**

$sql = "SELECT id, firstname, lastname FROM MyGuests";  
$result = $conn->query($sql);

To execute an SQL statement, you need first to store the statement in a string variable and then use $result = $conn->query($sql) to execute it, store the outcome in a variable (object) $result. Of course you may decide to use different names for the variables.

**Checking number of records**

if ($result->num\_rows > 0) {

//here you add the process

} else {  
    echo "0 results";

}

The above code is used to ensure that the result of the SQL statement contains at least one record. In case the result is not empty, then you go record by record and process the data.

**Processing records**

while($row = $result->fetch\_assoc()) {

        echo "id: " . $row["id"]. " - Name: " . $row["firstname"];

}

The above while loop goes through the results of the SQL statement and store each record in an object called $row. To get the value of each field, you need to use $row[‘filedname’].

**Closing Database connection:**

$conn->close();

Closing the database connection is simple. You need only to use the above statement.

**Passing variables to PhP page through URL**

<a href='page2.php?id=2489&user=tom'>link to page2</a>

The above code passes two variables to the php page page2.php. the variables are id and user.

**Process passed variable in the receiving page**

echo $\_GET['id']; // output 2489

echo $\_GET['tom']; // output tom

The above code is used in page2.php to process the variable received through URL. The object $\_GET[‘variablename’] is used to retrieve the values of the variable.

**Passing variables using a form using POST**

<form method="post" action=”page2.php”);?>">

To pass variable through a form, you need to add the URL of the targeted page in the form action and declare the method as “POST”.

Each input field of the form should have a name. For example,

<input type="text" name="firstname">

The last input line the form should be used for submit:

<input type="submit" name="submit" value="Submit">

**Retrieving variables that are sent through form post**

In the targeted page, in our case page2.php, you need to use $\_POST[‘variablename’] to retrieve the data stored in the form variable.

For example:

echo $\_POST[‘firstname’];

**Login page**

Create a form with user name and password.

<form action="login.php" method="post">

<div class="form-group">

<input class="form-control" id="name" name="username" placeholder="username" type="text">

<input class="form-control" id="password" name="password" placeholder="\*\*\*\*\*\*\*\*\*\*" type="password">

<input class="btn btn-default" name="submit" type="submit" value=" Login ">

</div>

</form>

The action section should include the php file that handles the login process. In this example, the file is login.php

At the beginning of the login page, you should include the following code:

<?php

include('login.php'); // Includes Login Script

require\_once('header.php');

if(isset($\_SESSION['login\_user'])){

header("location: profile.php");

}

**Session variables:**

Session variables are used to store information that are needed for the remaining pages in your project. The most important session variable is the user id or username. By storing the username in a session variable, you should be able to provide users with information specific to his/her account. In this tutorial we will use the session variable login\_user.

**Storing data in session variable**

To store data in a session variable you need to use the following line:

$\_SESSION['variable\_name']=$value;

In the example we use:

$\_SESSION['login\_user']=$username;

This means the value of $username will be stored in the session variable login\_user.

**Reading data from a session variable**

Reading data stored in a session variable is very simple. It is very similar to reading data from $\_GET and $\_POST. The only difference is using $\_SESSION instead of $\_GET and $\_POST.

In our example we use $\_SESSION['login\_user']

**Starting a session:**

Every page that is using session variables must start with this line:

session\_start(); // Starting Session

**Handling login process using login.php**

The file login.php performs the following:

* Start the session

session\_start(); // Starting Session

* Ensure that submit button of the form has been clicked and data is submitted through this button
* Check to make sure that the username or password are not empty

if (isset($\_POST['submit'])) {

if (empty($\_POST['username']) || empty($\_POST['password'])) {

$error = "Username or Password is invalid";

}

else

{

//login process

}

* Read data entered by the user from the input fields $username $password using $\_POST

$username=$\_POST['username'];

$password=$\_POST['password'];

* Create database connection and search for username and password. See previous section on how to connect to database and use SQL to search a table.
* If a record that is matching the same username and password, then store the username in the session variable login\_user and direct the page to the main page. In our example the main page is profile.php

**Handling sessions using session.php**

More advance sessions can be handled using a separate session.php code. In this example, we are using a database field named userType which stores the user type. Users types used in this example are AD for Admin, ST for student and FA for faculty.

The session.php performs the following functionalities:

* Connect to the database and search for records that has the same username value
* If successful, store the userType in a variable $userType

**Main page (profile.php)**

After successful login, the user is taken to a page that will provide him/her with specific information related to his/her account. In this example profile.php is used for this purpose. The profile.php performs the following process:

* include('session.php');
* Check the value stored in $userType
* If the value is “AD” then provide information and links specific to this file. The user of type AD will have full administration access.
* If the value is ST, the page will provide data specific to the student
* If the value is FA, the page will provide data specific to the faculty.
* The profile.php allows the user to logout using the file logout.php

**Logout**

The main purpose of logout.php is to delete all session variables and forward the user to the main login screen. This is done using by the following code:

<?php

session\_start();

if(session\_destroy()) // Destroying All Sessions

{

header("Location: index.php"); // Redirecting To Home Page

}

?>