How do you create session & cookie in PHP? Give difference between session and cookie with example. OR What are cookies? Explain the cookies handling in PHP with proper example.

PHP Sessions

- A PHP session variable is used to store information about, or change settings for a user session.
- Session variables hold information about one single user, and are available to all pages in one application.

PHP Session Variables

- When you are working with an application, you open it, do some changes and then you close it.
- This is much like a Session. The computer knows who you are. It knows when you start the application and when you end.
- But on the internet there is one problem: the web server does not know who you are and what you do because the HTTP address doesn't maintain state.
- A PHP session solves this problem by allowing you to store user information on the server for later use (i.e. username, shopping items, etc). However, session information is temporary and will be deleted after the user has left the website. If you need a permanent storage you may want to store the data in a database.
- Sessions work by creating a unique id (UID) for each visitor and store variables based on this UID. The UID is either stored in a cookie or is propagated in the URL.

Starting a PHP Session

- Before you can store user information in your PHP session, you must first start up the session.
- The session_start() function must appear BEFORE the <html> tag:

```
<?php session_start(); ?>
<html>
<body>
</body>
</html>
```

- The code above will register the user's session with the server, allow you to start saving user information, and assign a UID for that user's session.

Storing a Session Variable

- The correct way to store and retrieve session variables is to use the PHP $_SESSION variable:

```
<?php
session_start();
// store session data
$_SESSION['views']=1;
?>
<html>
<body>
<?php
//retrieve session data
echo "Pageviews=". $_SESSION['views'];
?>
</body>
</html>
```

- **Output**: Pageviews=1
- In the example below, we create a simple page-views counter. The isset() function checks if the "views" variable has already been set. If "views" has been set, we can increment our counter. If "views" doesn't exist, we create a "views" variable, and set it to 1:

```
<?php session_start();
if(isset($_SESSION['views']))
$_SESSION['views']=$_SESSION['views']+1;
else
$_SESSION['views']=1;
echo "Views=". $_SESSION['views'];
?>
```

Destroying a Session

- If you wish to delete some session data, you can use the unset() or the session_destroy() function.
- The unset() function is used to free the specified session variable:

```
<?php
unset($_SESSION['views']);
?>
```

- You can also completely destroy the session by calling the session_destroy() function.

```
<?php
session_destroy();
?>
```

- **Note**: session_destroy() will reset your session and you will lose all your stored session data.

PHP Cookie

- A cookie is often used to identify a user. A cookie is a small file that the server embeds on the user's computer.
- Each time the same computer requests a page with a browser, it will send the cookie too. With PHP, you can both create and retrieve cookie values.

How to Create a Cookie?

- The setcookie() function is used to set a cookie.
- **Note**: The setcookie() function must appear BEFORE the <html> tag.

```
setcookie(name, value, expire, path, domain);
```

- In the example below, we will create a cookie named "user" and assign the value "Alex Porter" to it. We also specify that the cookie should expire after one hour:

```
<?php
setcookie("user", "Alex Porter", time()+3600);
?>
```

- **Note**: The value of the cookie is automatically URLencoded when sending the cookie, and automatically decoded when received (to prevent URLencoding, use setrawcookie() instead).
- You can also set the expiration time of the cookie in another way. It may be easier than using seconds.

```
<?php
$expire=time()+60*60*24*30;
setcookie("user", "Alex Porter", $expire);
?>
```

- In the example above the expiration time is set to a month (60 sec * 60 min * 24 hours * 30 days).

How to Retrieve a Cookie Value?

- The PHP $_COOKIE variable is used to retrieve a cookie value.
- In the example below, we retrieve the value of the cookie named "user" and display it on a page:

```
<?php
// Print a cookie
echo $_COOKIE["user"];

// A way to view all cookies
print_r($_COOKIE);
?>
```

In the following example we use the isset() function to find out if a cookie has been set:

```
<html>
<body>

<?php
if (isset($_COOKIE["user"]))
  echo "Welcome " . $_COOKIE["user"] . "!<br />";
else
  echo "Welcome guest!<br />";
?>

</body>
</html>
```

How to Delete a Cookie?

- When deleting a cookie you should assure that the expiration date is in the past.

```
<?php
// set the expiration date to one hour ago
setcookie("user", "", time()-3600);
?>
```

What if a Browser Does NOT Support Cookies?

- If your application deals with browsers that do not support cookies, you will have to use other methods to pass information from one page to another in your application. One method is to pass the data through forms (forms and user input are described earlier in this tutorial).\
- The form below passes the user input to "welcome.php" when the user clicks on the "Submit" button:

```
<html>
<body>

<form action="welcome.php" method="post">
Name: <input type="text" name="name" />
Age: <input type="text" name="age" />
<input type="submit" />
</form>

</body>
</html>
```

- Retrieve the values in the "welcome.php" file like this:

```
<html>
<body>

Welcome <?php echo $_POST["name"]; ?>.<br />
You are <?php echo $_POST["age"]; ?> years old.

</body>
</html>
```