Assignment: 3

1. **What is cookie? Explain attributes of cookie. Explain with example(program).**

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.

Attribute of cookie:

Cookies are pieces of information stored on the client side, which are sent to the server with every request made by the client. Cookies are primarily used for authentication and maintaining sessions. Hence, securing a cookie effectively means securing a user's identity. Cookies can be secured by properly setting cookie attributes.

- Secure
- Domain
- Path
- HTTP Only
- Expires

  - **Secure: -** The 'Secure' attribute makes sure that the cookie will only be sent with requests made over an encrypted connection and an attacker won't be able to steal cookies by sniffing. However, we need to be very careful while setting this attribute. Just setting the attribute to 'Secure' does not necessarily mean that the cookie will always be transmitted over an encrypted connection. RFC 2965 states, *When it sends a "secure" cookie back to a server, the user agent SHOULD use no less than the same level of security as was used when it received the cookie from the server.*

- **Domain: -** The '**domain**' attribute signifies the domain for which the cookie is valid and can be submitted with every request for this domain or its subdomains. If this attribute is not specified, then the hostname of the originating server is used as the default value.
- **Path: -** The '**path**' attribute signifies the URL or path for which the cookie is valid. The default path attribute is set as '/'.
- **HTTP only: -** this attribute is used to cross Site Scripting attacks can be used to steal cookies with the help of client-side scripts.
- **Expires: -** This attribute is used to set persistent cookies. It signifies how long the browser should use the persistent cookie and when the cookie should be deleted.

EXAMPLE

```
<html>
<body>

<?php
if(!isset($_COOKIE[$cookie_name])) {
  echo "Cookie named '" . $cookie_name . "' is not set!";
} else {
  echo "Cookie '" . $cookie_name . "' is set!<br>";
  echo "Value is: " . $_COOKIE[$cookie_name];
}
?>

</body>
</html>

<?php
$cookie_name = "user";
$cookie_value = "John Doe";
setcookie($cookie_name, $cookie_value, time() + (86400 * 30), "/"); // 86400 = 1 day
?>
```

2. What is session? Explain attributes of session. Explain with example(program).

When you work 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 or what you do, because the HTTP address doesn't maintain state.

Session variables solve this problem by storing user information to be used across multiple pages (e.g. username, favorite color, etc.). By default, session variables last until the user closes the browser.

So; Session variables hold information about one single user, and are available to all pages in one application.

Attributes of session:

- **Starting a PHP Session**: The first step is to start up a session. After a session is started, session variables can be created to store information. The PHP **session_start()** function is used to begin a new session. It also creates a new session ID for the user.
  Below is the PHP code to start a new session:

  filter_none
  brightness_4

```
<?php

session_start();

?>
```

- **Storing Session Data**: Session data in key-value pairs using the **$_SESSION[]** super global array. The stored data can be accessed during lifetime of a session.
  Below is the PHP code to store a session with two session variables Roll number and Name:

  filter_none
  brightness_4

```
<?php

session_start();
```

```
$_SESSION["Rollnumber"] = "37";
$_SESSION["Name"] = "Meet";

?>
```

- **Accessing Session Data**: Data stored in sessions can be easily accessed by firstly calling **session_start()** and then by passing the corresponding key to the **$_SESSION** associative array.
  The PHP code to access a session data with two session variables Roll number and Name is shown below:

```
filter_none
brightness_4

<?php

session_start();

echo 'The Name of the student is:' . $_SESSION["Name"] . '<br>';
echo 'The Roll number of the student is :' . $_SESSION["Rollnumber"] . '<br>';

?>
```

Output:

```
The Name of the student is: Meet

The Roll number of the student is :37
```

Destroying Certain Session Data: To delete only a certain session data, the unset feature can be used with the corresponding session variable in the **$_SESSION** associative array.
The PHP code to unset only the “Rollnumber” session variable from the associative session array:

```
filter_none
brightness_4

<?php

session_start();

if(isset($_SESSION["Name"])){
   unset($_SESSION["Rollnumber"]);
}

?>
```

- **Destroying Complete Session**: The **session_destroy()** function is used to completely destroy a session. The session_destroy() function does not require any argument.
  filter_none
  brightness_4

  ```
  <?php

  session_start();
  session_destroy();

  ?>
  ```

EXAMPLE:

```
<!DOCTYPE html>
<html>
<body>

<?php
// Set session variables
$_SESSION["favcolor"] = "green";
$_SESSION["favanimal"] = "cat";
echo "Session variables are set.";
?>

</body>
</html>
```

```
<?php
// Start the session
session_start();
?>
```