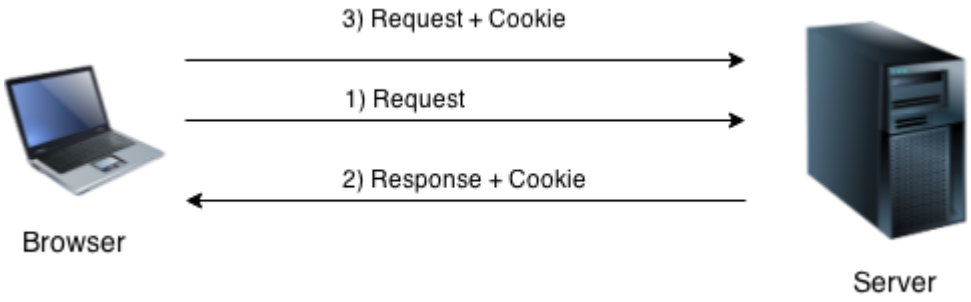
PHP Cookie

A **cookie** in PHP is a small file with a maximum size of 4KB that the web server stores on the client computer.

They are used to store information about the user's visit to the website, such as their preferences, login information, and shopping cart contents. This information can then be used to personalize the user's experience on the website. It is used to recognize the user.

A cookie can only be read from the domain that it has been issued from. Cookies are usually set in an HTTP header but JavaScript can also set a cookie directly on a browser.

Cookie is created at server side and saved to client browser. Each time when client sends request to the server, cookie is embedded with request. Such way, cookie can be received at the server side.



In short, cookie can be created, sent and received at server end.

> Note: PHP Cookie must be used before <html> tag.

Creating a Cookie: setcookie() function

PHP setcookie() function is used to set cookie with HTTP response. Once cookie is set, you can access it by $_COOKIE superglobal variable.

Syntax:

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

Parameters: The setcookie() function requires six parameters in general which are:

- **Name:** It is used to set the name of the cookie.

- **Value:** It is used to set the value of the cookie.
- **Expire:** It is used to set the expiry timestamp of the cookie after which the cookie can't be accessed.
- **Path:** It is used to specify the path on the server for which the cookie will be available.Ex: "/mypath/"
- **Domain:** It is used to specify the domain for which the cookie is available. Ex: "mydomain.com"
- **Security:** It is used to indicate that the cookie should be sent only if a secure HTTPS connection exists. Ex: 1.

Example:

cookies.php

```
 <?php
  setcookie('username','Arohi',time()+10);
?>
```

The above code will create a cookie named `'username` with the value `'Arohi'` that will expire after `10` second.

Note: Only the name argument in the setcookie() function is mandatory. To skip an argument, the argument can be replaced by an empty string("").

Accessing Cookie Values: PHP $_COOKIE

For accessing a cookie value, the PHP $_COOKIE superglobal variable is used. It is an associative array that contains all the cookies values sent by the browser in the current request. The key of the array is the name of the cookie, and the value of the array is the value of the cookie.

Example:

veiw.php

```
<?php
```

```
  echo "COOKIES['username']: ".$_COOKIE['username'];
?>
```

The above code will retrieve the value of the 'username' cookie.

Checking Whether a Cookie Is Set Or Not: It is always advisable to check whether a cookie is set or not before accessing its value. Therefore to check whether a cookie is set or not, the PHP isset() function is used.

Example:

```
<?php
  // echo "COOKIES['username']: ".$_COOKIE['username'];
  if(isset($_COOKIE['username']))
  {
    echo "COOKIES['username']: ".$_COOKIE['username'];
  }
  else
  {
    echo "Sorry, cookie is not found!";
  }
?>
```

PHP Delete Cookie

To delete a cookie, you can set its expiration time to a time in the past. For deleting a cookie, the setcookie() function is called by passing the cookie name and other arguments or empty strings but however this time, the expiration time is required to be set in the past.

cookie.php

```
<?php
  setcookie('username','',time()-10);// set the expiration time to 10 sec ago
?>
```

Important Points:

I. If the expiration time of the cookie is set to 0 or omitted, the cookie will expire at the end of the session i.e. when the browser closes.

II. The same path, domain, and other arguments should be passed that were used to create the cookie in order to ensure that the correct cookie is deleted.

Example: Remembering a User's Preferences

The following code shows how to use cookies to remember a user's preferred color:

```php
<?php
 if (isset($_COOKIE["color"]))
 {
   $color = $_COOKIE["color"];
 }
 else
 {
   $color = "blue";
 }

 echo "<style>";
 echo "body { background-color: $color; }";
 echo "</style>";
?>
```

This code will check to see if the color cookie is set. If it is, then the user's preferred color will be used to style the website. Otherwise, the default color of blue will be used.

Example: Creating a Shopping Cart

```php
<?php
if (isset($_POST["item"])) {
  $item = $_POST["item"];
  if (isset($_COOKIE["cart"])) {
    $cart = json_decode($_COOKIE["cart"], true);
  } else {
    $cart = [];
  }

  if (!in_array($item, $cart)) {
    array_push($cart, $item);
  }

  setcookie("cart", json_encode($cart), time() + 3600);
}
```

```php
echo "<ul>";
foreach ($_COOKIE["cart"] as $item) {
  echo "<li>$item</li>";
}
echo "</ul>";
?>
```

This code will check to see if the user has added an item to their shopping cart. If they have, then the item will be added to the $_COOKIE["cart"] array. The $_COOKIE["cart"] array is a JSON-encoded string that contains the list of items in the shopping cart.

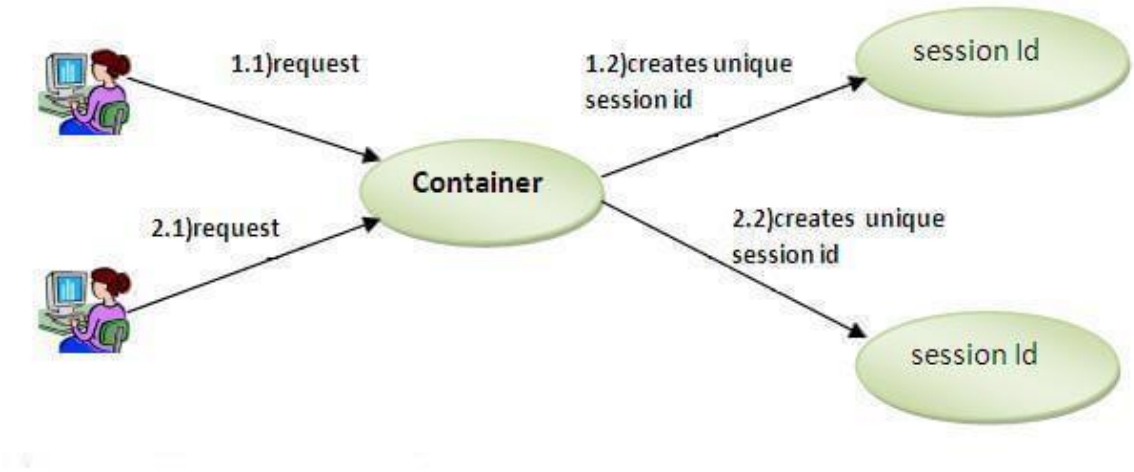
PHP Session

PHP session is used to store and pass information from one page to another temporarily (until user close the website).

PHP session technique is widely used in shopping websites where we need to store and pass cart information e.g. username, product code, product name, product price etc from one page to another.

A PHP session is used to store data on a server rather than the computer of the user. his means that the information is persistent, even if the user closes their browser and comes back to the website later.

Session identifiers or SID is a unique number which is used to identify every user in a session based environment. The SID is used to link the user with his information on the server like posts, emails etc. PHP session creates unique user id for each browser to recognize the user and avoid conflict between multiple browsers.



How are sessions better than cookies?

Although cookies are also used for storing user related data, they have serious security issues because cookies are stored on the user's computer and thus they are open to attackers to easily modify the content of the cookie. Addition of harmful data by the attackers in the cookie may result in the breakdown of the application.

Apart from that cookies affect the performance of a site since cookies send the user data each time the user views a page. Every time the browser requests a URL to the server, all the cookie data for that website is automatically sent to the server within the request.

PHP session_start() function

PHP session_start() function is used to start the session. It starts a new or resumes existing session. It returns existing session if session is created already. If session is not available, it creates and returns new session. After a session is started, session variables can be created to store information.

Example:

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

Storing Session Data: PHP $_SESSION

PHP $_SESSION is an associative array that contains all session variables. It is used to set and get session variable values. The stored data can be accessed during lifetime of a session.

Example: Store information

```php
<?php
  session_start();
  $_SESSION['username']='Arohi';
?>
```

Storing Session Data: PHP $_SESSION

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.

Example: Get information

```php
<?php
  session_start();
  echo "SESSION['username']: ".$_SESSION['username'];
?>
```

PHP Session Counter Example

File: sessioncounter.php

```php
<?php
  session_start();

  if (!isset($_SESSION['counter'])) {
    $_SESSION['counter'] = 1;
  } else {
    $_SESSION['counter']++;
  }
  echo ("Page Views: ".$_SESSION['counter']);
?>
```

PHP Destroying Complete Session

PHP session_destroy() function is used to destroy all session variables completely.

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

PHP Destroying Certain Session

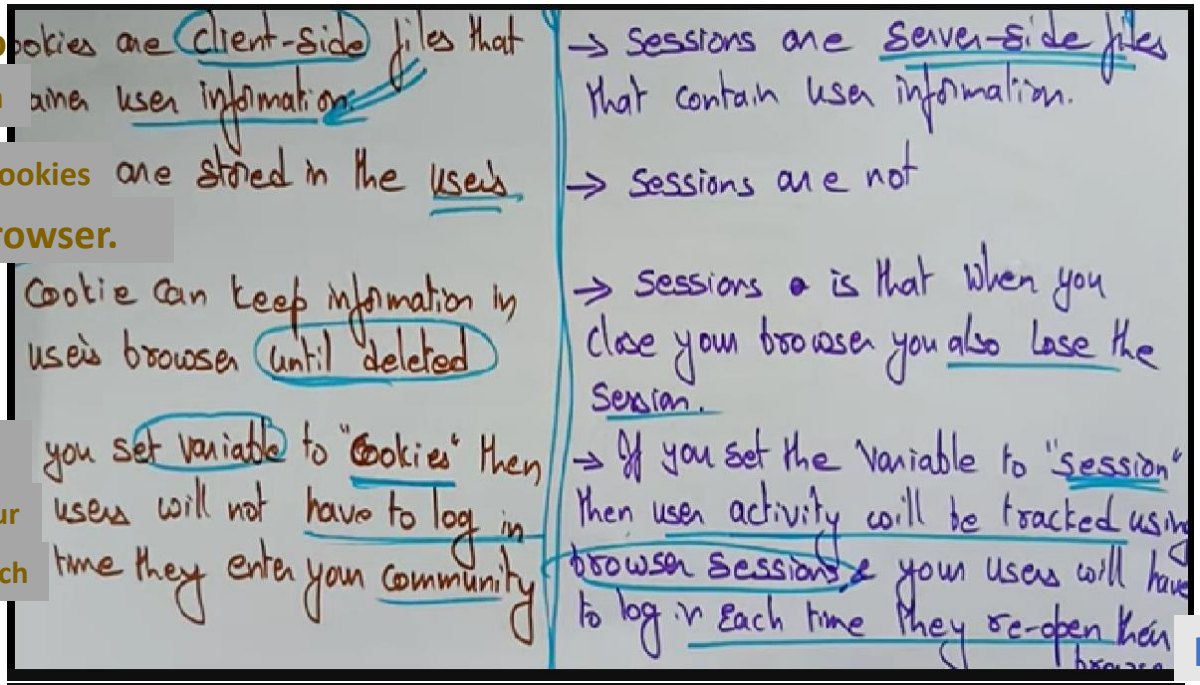
To delete only a certain session data, the unset feature can be used with the corresponding session variable in the **$_SESSION** associative array.

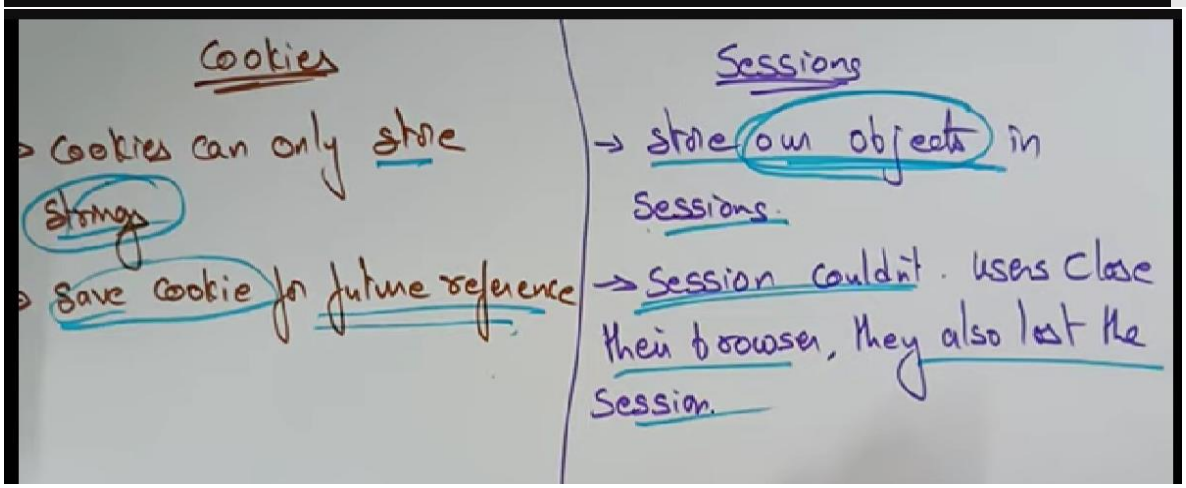
The PHP code to unset only the “Rollnumber” session variable from the associative session array:

```php
<?php

session_start();

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

?>
```

Important Points

1. The session IDs are randomly generated by the PHP engine .
2. The session data is stored on the server therefore it doesn’t have to be sent with every browser request.
3. The session_start() function needs to be called at the beginning of the page, before any output is generated by the script in the browser.


cookies are client-side files that contain user information.
cookies are stored in the user's browser.
Cookie can keep information in user's browser until deleted
If you set variable to "Cookies" then your users will not have to log in each time they enter your community
→ sessions are server-side files that contain user information.
→ sessions are not
→ sessions • is that when you close your browser you also lose the session.
→ If you set the variable to "session" then user activity will be tracked using browser sessions & your users will have to log in each time they re-open their browser


Cookies
Cookies can only store strings
Save cookie for future reference
Sessions
→ store our objects in Sessions.
→ Session couldn't. Users close their browser, they also lost the session.