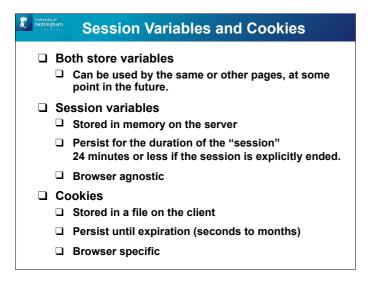


University of Nottingha	GET vs POST
:	HTML forms are used to submit user data to PHP scripts
u i	Different HTTP request methods
	GET passes data on the URL
	POST passes data from the server to the browser internally
	☐ GET requests can be cached, bookmarked and remain in the browser history POST does not
Į į	GET requests have length restrictions, POST does not
Į (	GET requests are visible to everyone and so easy to reverse engineer
	POST requests are invisible so more secure

Univer Nott	sity of ingham	Be aware of security issues
		nere are potential PHP "exploits" - especially Cross Site Scripting" (XSS)
	F	orm Validation makes this much less likely
		arious PHP functions can be used to process ser input and prevent attacks
		Use regular expressions to validate data items
		htmlspecialchars() returns a string with special HTML characters (e.g. <) converted to entities (e.g. \$lt;).
		trim() removes unnecessary whitespace
		stripslashes() removes backslashes
	Se htt	ee: ps://www.w3schools.com/php/php_form_validation.asp



```
Reading a cookie

$cookie_name = "person";

if(!isset($_COOKIE[$cookie_name]))
{
    ... cookie not set
}
    else
{
    echo "Value: " . $_COOKIE[$cookie_name];
}
```

```
c?php
$cookie_name = "person";
setcookie($cookie_name, null, -1);
?>
```

```
Cookie issues

Intended for long term storage of information (e.g. personal preferences or user profiles)

Not compatible across browsers

Cookies can be disabled by browsers
Before using cookies, check that they are working!

Moderately but not totally secure
It is bad practice to store sensitive information (e.g. passwords) in cookies
```

```
Use session_start() function to start a new session

| NB you can only have one session at once
| This will last for 24 minutes, unless actively destroyed
| session_start must be executed in each PHP file that uses session variables

| Use the $_SESSION superglobal to assign and read variables
| This is an associative array - the session variable 'myVar' is thus $_SESSION['myVar']

| A session can be ended using session_destroy()
| This destroys all session variables
| NB you must do a session_start() before you can do a session_destroy()
```

```
create a new session

create a ne
```

