Índex de continguts

1 Seguretat i control d'errors

Moltes vegades és important verificar la identitat dels dos extrems d'una comunicació, hi ha mètodes per identificar tant al servidor en el qual s'allotja el lloc web, com a l'usuari del navegador que es troba en l'altre extrem.

Els llocs web que necessiten emprar identificació de servidor, com les botigues o els bancs, utilitzen el PROTOCOL HTTPS. Aquest protocol requereix d'un certificat digital vàlid, signat per una autoritat fiable, que és verificat pel navegador quan s'accedeix a la pàgina web. A més, HTTPS utilitza mètodes de xifrat per crear un canal segur entre el navegador i el servidor, de tal manera que no es puga interceptar la informació que es transmet pel mateix.

Per identificar els usuaris que visiten un lloc web, es poden utilitzar diferents mètodes com el DNI digital o certificats digitals d'usuari (document digital que conté informació sobre l'usuari com el nom o l'adreça. Aquesta informació està signada per una altra entitat, anomenada entitat certificadora, que ha de ser de confiança i garanteix que la informació que conté és certa), però el més estès és sol·licitar a l'usuari certa informació que només ell coneix: la combinació d'un nom d'usuari i una contrasenya.

En les unitats anteriors vas aprendre a utilitzar aplicacions web per gestionar informació emmagatzemada en bases de dades. En la majoria dels casos és important implantar en aquest tipus de aplicacions web, les que accedeixen a bases de dades, algun mecanisme de control d'accés que obliga a l'usuari a identificar-se. Un cop identificat, es pot limitar l'ús que pot fer de la informació.

Així, pot haver llocs web en els quals els usuaris autenticats poden utilitzar només una part de la informació (com els bancs, que permeten als seus clients accedir únicament a la informació relativa als seus comptes). Altres llocs web necessiten separar en grups als usuaris autentificats, de tal manera que la informació a la qual accedeix un usuari depèn del grup en què aquest es trobe. Per exemple, una aplicació de gestió d'una empresa pot tenir un grup d'usuaris als qui permet visualitzar la informació, i un altre grup d'usuaris que, a més de visualitzar la informació, també la poden modificar.

> Has distingir l'autenticació dels usuaris i el control d'accés, de la utilització de mecanismes per assegurar les comunicacions entre l'usuari de el navegador i el servidor web. Encara que tots dos aspectes solen anar units, són independents.

En els exemples d'aquesta unitat, la informació d'autenticació (nom i contrasenya dels usuaris) s'envia en text pla des del navegador fins al servidor web. Aquesta pràctica és altament insegura i mai s'ha d'usar sense un protocol com HTTPS que permeta xifrar les comunicacions amb el lloc web. No obstant això,

la configuració de servidors web que permeten fer servir el protocol HTTPS per xifrar la informació que reben i transmeten no forma part dels continguts d'aquest mòdul. Per aquest motiu, durant aquesta unitat utilitzarem únicament el protocol no segur HTTP.

1.1 Manteniment de l'estat

Recorda que HTTP és un protocol sense estat, el que significa que un cop un servidor web completa la sol·licitud d'un client d'un recurs, la connexió entre els dos acaba. Dit d'una altra manera, no hi ha manera que un servidor recorde una que una seqüència de sol·licituds prové del mateix client.

> Mantenir l'estat és poder fer el seguiment d'una sequència de sol·licituds d'un client.

No obstant això mantenir l'estat, poder fer el seguiment d'una seqüència de sol·licituds d'un client, és molt útil. No podeu crear una aplicació que tinga cistella de la compra, per exemple, si no podeu mantenir l'estat. Heu de saber quan un usuari afegeix articles a la cistella o els elimina i el contingut final de la cistella si el client decideix adquirir els productes.

Per solucionar aquesta manca d'estat de la web tenim disponibles diverses tècniques com l'ús cookies o de sessions que veurem a continuació.

1.2 Combinant mecanismes d'autenticació i tècniques per a mantenir l'estat podem crear aplicacions web segures.

layout: default title: 1. Galetes (cookies) nav_order: 1 parent: 8. Seguretat i control d'errors —

2 Galetes (*cookies*)

Una galeta és un fitxer de text que un lloc web guarda a l'entorn de l'usuari de navegador. El seu ús més típic és l'emmagatzematge de les preferències de l'usuari (per exemple, l'idioma en que s'han de mostrar les pàgines), perquè no hagi de tornar a indicar-les la propera vegada que visiteu el lloc.

Si utilitzes Firefox com a navegador, pots accedir a `Desenvolupador web – Inspector d' emmagatzematge` des del menú principal. Entre les seves característiques et permet consultar i editar les galetes emmagatzemades en el mateix.

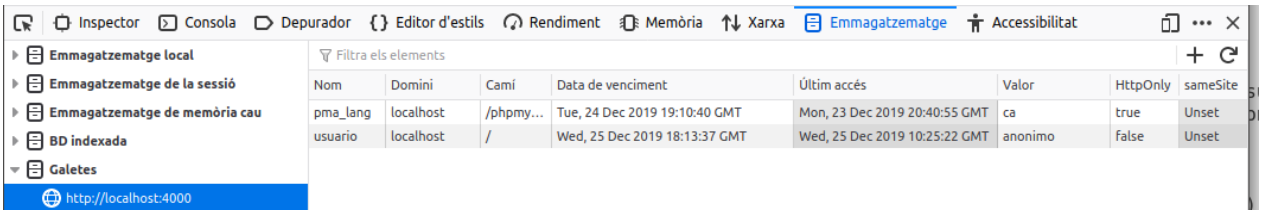

Figure 1: Inspeccionar galetes en Firefox

En PHP, per emmagatzemar una galeta al navegador de l'usuari, pots utilitzar la funció `setcookie`. L'únic paràmetre obligatori que has de fer servir és el nom de la galeta, però admet diversos paràmetres més opcionals.

```
setcookie ($name [,
    $value = " " [,
    $expires = 0 [,
    $path = "" [,
    $domain = "" [, $secure = FALSE [,
    $httponly = FALSE ]]]]]] ) : bool
```

Per a més informació consulta: https://www.php.net/manual/es/function.setcookie.php.

Per exemple, si vols emmagatzemar en una galeta la data del darrer accés d'un usuari, pots fer:

```
setcookie ("last_visit_date", (string) time(), time() + 3600);
```

Els dos primers paràmetres són el nom de la galeta i el seu valor. El tercer és la data de caducitat de la mateixa en format `timestamp` (en l'exemple, un hora des del moment en què s'execute). En cas de no figurar aquest paràmetre, la galeta s'eliminarà quan es tanque el navegador. Tingues en compte que també es poden aplicar restriccions a les pàgines del lloc que poden accedir a una galeta en funció de la ruta.

Les galetes es transmeten entre el navegador i el servidor web utilitzant les capçaleres del protocol HTTP. PER AIXÒ, LES SENTÈNCIES `setcookie` S'HAN D'ENVIAR ABANS QUE EL NAVEGADOR MOSTRE CAP INFORMACIÓ A PANTALLA.

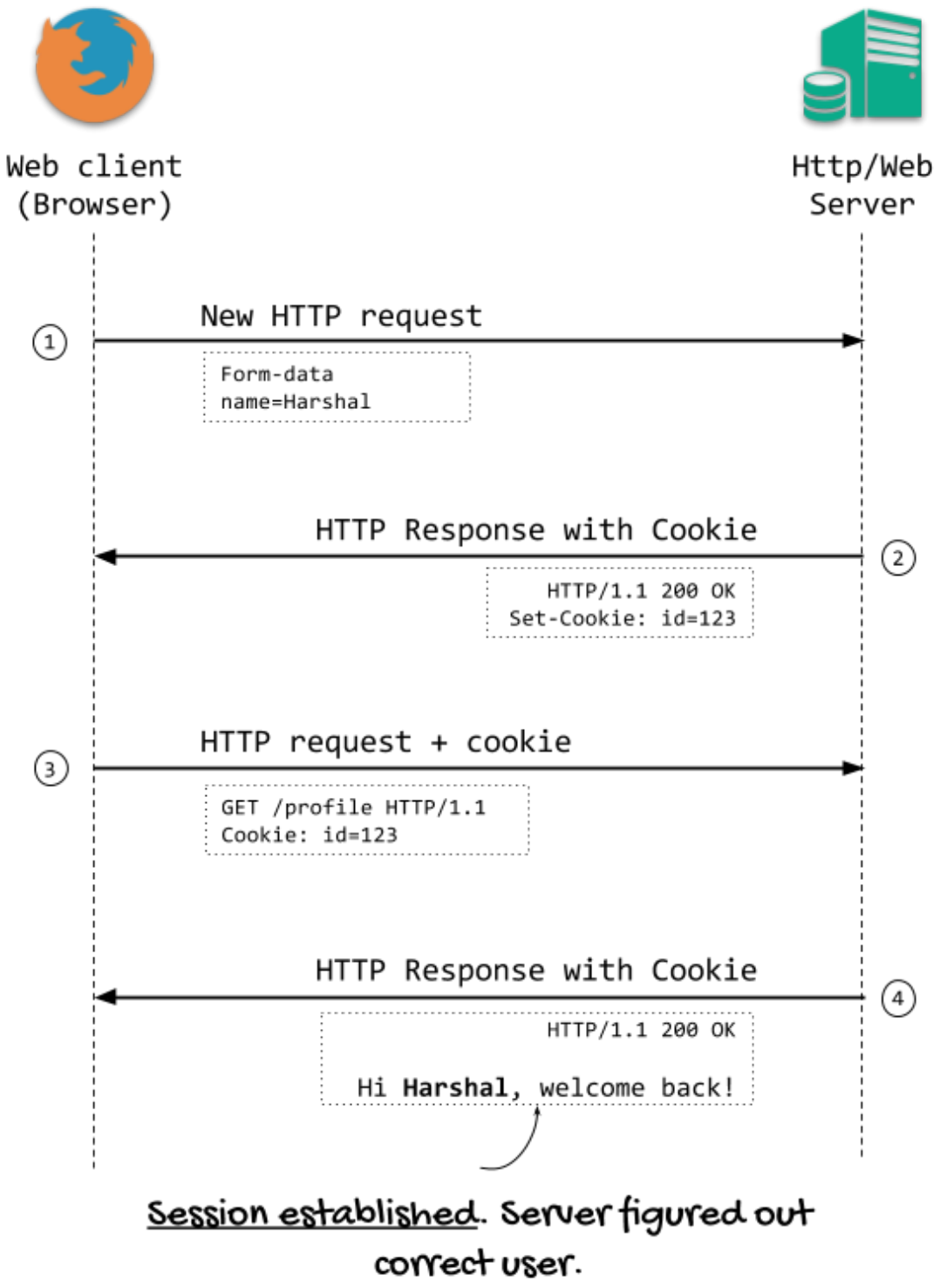

Figure 2: Enviament de cookies HTTP

Podeu trobar més informació en Ultimate Guide to HTTP Cookies

El procés de recuperació de la informació que emmagatzema una galeta és molt simple. quan accedeixes a un lloc web, el navegador li envia de forma automàtica tot el contingut de les galetes que emmagatzema relatives a aquest lloc en concret. Des PHP pots accedir a aquesta informació per mitjà de l'array `$_COOKIE`.

> Sempre que utilitzeu galetes en una aplicació web, heu de tenir en compte que en última instància la seva disponibilitat està controlada pel client. Per exemple, alguns usuaris deshabiliten les galetes al navegador perquè pensen que la informació que emmagatzemen pot suposar un potencial problema de seguretat. O que la informació que emmagatzemen pot arribar a perdre perquè el usuari decideix formatar l'equip o simplement eliminar-les de sistema.

Si un cop emmagatzemada una galeta al navegador vols eliminar-la abans que expire, pots utilitzar la mateixa funció `setcookie` però indicant una data de caducitat anterior a l'actual.

Exercici pràctic

Activitat 1: Última visita

Modifica la pàgina `index.php` de forma que emmagatzeme en una galeta l'últim instant en què l'usuari va visitar la pàgina.
Si és la seva primera visita, mostra un missatge de benvinguda. En cas contrari, mostra la data i hora de la seva anterior visita.
La galeta tindrà una durada d'una setmana.
Comprova que la galeta s'ha creat correctament.

Solució proposada

```php
declare(strict_types=1);

// it's a good practise to save the cookie name in a variable
// to avoid misspelling errors.
$cookieName = "last_visit_date";

// we get the current cookie value
$lastVisit = filter_input(INPUT_COOKIE, $cookieName, FILTER_VALIDATE_INT);

// we can also use the coalsece operator
/* $lastVisit =(int)($_COOKIE[$cookieName] ?? null); //
// or the traditional isset
```

```php
/* if (isset($_COOKIE[$cookieName])) {
    $lastVisit = (int)$_COOKIE[$cookieName];
} else
    $lastVisit = null;
*/

// if null we show a welcome message
if (empty($lastVisit))
    $message = "Welcome to our reservation system!";
else
    $message = "Welcome back, your last visit was on " .
        date("d/m/Y h:i:s", $lastVisit);

// we register the current time and set the expiration date to the next
   week.
setcookie($cookieName, (string)time(), time() + 7 * 24 * 60 * 60);
```

Exercici pràctic

Activitat 2. Emmagatzematge del nom de l'usuari

L'objectiu és que quan un usuari faça una reserva s'emmagatzeme el nom introduït en un galeta, de forma que quan torne a accedir al formulari de reserva el nom aparega automàticament.
El nom de la *cookie* serà `last_used_name` i la durada serà de 30 dies.
Nota: l'estructura és semblant a l'anterior però estarà divida en dues pàgines.
Crea la pàgina `logout.php` que en accedir eliminarà la *cookie*.

Quina és la durada per defecte d'una galeta si no s'indica la data de caducitat, com en la següent crida a la funció `setcookie`?

```php
setcookie ("idioma", "espanyol");
```

1. Fins que es tanque el navegador de l'usuari.
2. 1 hora.

3 Maneig de sessions

3.1 Introducció

Com acabes de veure, una forma per guardar informació particular de cada usuari és utilitzar galetes (*cookies*). No obstant això, hi ha diversos problemes associats a les galetes, com el nombre d'elles que admet el navegador, o la seva grandària màxima. Per solucionar aquests inconvenients, existeixen *les sessions*. El terme sessió fa referència al conjunt d'informació relativa a un usuari concret.

Aquesta informació pot ser tan simple com el nom de l'usuari mateix, o més complexa, com els articles que ha dipositat a la cistella de compra d'una botiga en línia.

3.2 Identicadors de sessió

Cada usuari diferent d'un lloc web té la seva pròpia informació de sessió. Per distingir una sessió d'una altra s'usen els IDENTIFICADORS DE SESSIÓ (SID). Un SID és un atribut que s'assigna a cada un dels visitants d'un lloc web i l'identifica. D'aquesta manera, si el servidor web coneix el SID d'un usuari, pot relacionar-lo amb tota la informació que posseeix sobre ell, que es manté en la sessió de l'usuari. Aquesta informació s'emmagatzema en el servidor web, generalment en fitxers tot i que també es poden utilitzar altres mecanismes d'emmagatzematge com bases de dades. Com ja hauràs suposat, la qüestió ara és: ¿i on s'emmagatzema aquest SID, identificador de la sessió, que és únic per a cada usuari? Doncs hi ha dues maneres de mantenir el SID entre les pàgines d'un lloc web que visita l'usuari:

- Utilitzant galetes.
- Propagant el SID en un paràmetre de la URL. El SID s'afegeix com una part més de la URL, de la forma:
  - https://www.example.com/stor/list.php&PHPSESSID=34534FG4FFG34TY

En l'exemple anterior, el SID és el valor del paràmetre PHPSESSID.

Cap de les dues maneres és perfecta. Ja saps els problemes que té la utilització de cookies. Malgrat això, és el millor mètode i el més utilitzat. Propagar el SID com a part de la URL comporta majors desavantatges, com la impossibilitat de mantenir el SID entre diferents sessions, o el fet que compartir la URL amb una altra persona implica compartir també l'identificador de sessió.

3.3 Maneig de sessions en PHP

La bona notícia, és que el procés de maneig de sessions en PHP està automatitzat en gran mida. Quan un usuari visita un lloc web, no cal programar un procediment per veure si hi ha un SID previ i carregar les dades associades amb el mateix. Tampoc has d'utilitzar la funció `setcookie` si vols emmagatzemar els SID en galetes, o anar passant el SID entre les pàgines web del teu lloc si et decideixes per propagar. Tot això PHP ho fa automàticament.

> La informació que s'emmagatzema en la sessió d'un usuari també es coneix com galetes en la part de servidor (*server side cookies*). Has de tenir en compte que encara aquesta informació no viatja entre el client i el servidor, sí que ho fa el SID, bé com a part de l'URL o en una capçalera HTTP si es guarda en una galeta. En tots dos casos, això planteja un possible problema de seguretat. El SID pot ser aconseguit per una altra persona, i a partir de la mateixa obtenir la informació de la sessió de l'usuari. La manera més segura d'utilitzar sessions és emmagatzemant els SID en galetes i utilitzar HTTPS per a xifrar la informació que es transmet entre el servidor web i el client.

3.4 Configuració

Per defecte, PHP inclou suport de sessions incorporat. Abans, però, d'utilitzar sessions en el teu lloc web, has de configurar correctament PHP utilitzant els següents directives en el fitxer `php.ini` segons corresponga:

Directiva	significat
session.use_cookies	Indica si s'han d'usar cookies (1) o propagació a la URL (0) per emmagatzemar el SID.
session.use_only_cookies	S'ha d'activar (1) quan fas servir cookies per emmagatzemar els SID, i a més no vols que es reconeguin els SID que es puguin passar com part de la URL (aquest mètode es pot usar per usurpar l'identificador d'un altre usuari).

Directiva	significat
session.save_handler	S'utilitza per indicar a PHP com ha de emmagatzemar les dades de la sessió de l'usuari. Hi ha quatre opcions: en fitxers (files), en memòria (Mm), en una base de dades SQLite (sqlite) o utilitzant per a això funcions que ha de definir el programador (user). El valor per defecte (Files) funcionarà sense problemes en la majoria dels casos.
session.name	Determina el nom de la galeta que s'utilitzarà per guardar el SID. La seva valor per defecte és PHPSESSID.
session.auto_start	El seu valor per defecte és 0, i en aquest cas hauràs de fer servir la funció session_start per gestionar l'inici de les sessions. Si fas servir sessions al lloc web, pot ser bona idea canviar el seu valor a 1 per que PHP activi de forma automàtica el maneig de sessions.
session.cookie_lifetime	Si utilitzes l'URL per propagar el SID, aquest es perdrà quan tanqui el navegador. No obstant això, si utilitzes galetes, el SID es mantindrà mentre no es destrueixi la galeta. En el seu valor per defecte (0), les galetes es destrueixen quan es tanca el navegador. Si vols que es mantingui el SID durant més temps, has d'indicar en aquesta directiva aquest temps en segons.
session.gc_maxlifetime	Indica el temps en segons que s'ha de mantenir activa la sessió, encara que no hi hagi cap activitat per part de l'usuari. El seu valor per defecte és 1440. És a dir, passats 24 minuts des de l'última activitat per part de l'usuari, es tanca la sessió automàticament.
session.cookie_path	URL path prefix that must match for the cookie to be sent.

Directiva	significat
session.cookie_domain	Domain suffix that must match for the cookie to be sent. No value means the cookie is sent back only to the full hostname that sent it.
session.cookie_secure	Set to On to have the cookie only sent back with HTTPS URLs.
session.cookie_httponly	Set to On to tell browsers to prevent JavaScript from reading the cookie.

La funció `phpinfo`, de la qual ja vam parlar amb anterioritat, t'ofereix informació sobre la configuració actual de les directives de sessió.

En la documentació de PHP tens informació sobre aquestes i altres directives que permeten configurar el maneig de sessions.

https://www.php.net/manual/es/session.configuration.php

Si la informació de l'usuari que vols emmagatzemar inclou contingut privat com una contrasenya, ¿què utilitzaries, galetes o la sessió de l'usuari?

3.5 Inici i fi d'una sessió

L'inici d'una sessió pot tenir lloc de dues maneres. Si has activat la directiva `session.auto_start` en la configuració de PHP, la sessió començarà automàticament quan un usuari es connecte al teu lloc web. En el cas que aquest usuari ja haja obert una sessió amb anterioritat, i aquesta no s'haja eliminat, en lloc d'obrir una nova sessió es reprendrà la anterior. Per a això s'utilitzarà el SID anterior, que estarà emmagatzemat en una galeta (recorda que si fas servir propagació de l'SID, no podràs restaurar sessions anteriors; el SID figura a la URL i es perd quan tanques el navegador).

Si per contra, decideixes no utilitzar l'inici automàtic de sessions, hauràs executar la funció `session_start()` per indicar a PHP que inicie una nova sessió o reprenga l'anterior. Anteriorment aquesta funció tornava sempre **`true`**, a partir de la versió 5.3.0 de PHP el seu comportament és més coherent: retorna **`false`** en cas de no poder iniciar o restaurar la sessió.

> Com l'inici de sessió requereix utilitzar `cookies`, i aquestes es transmeten en els encapçalats HTTP, heu de tenir en compte que per poder iniciar una sessió utilitzant `session_start`, hauràs de fer les cridades a aquesta funció abans que la pàgina web mostre informació al navegador.

A més, totes les pàgines que necessiten utilitzar la informació emmagatzemada en la sessió, hauran d'executar la funció `session_start`.

Mentre la sessió estiga oberta, pots utilitzar la variable superglobal `$_SESSION` per afegir informació a la sessió de l'usuari, o per accedir a la informació emmagatzemada en la sessió. Per exemple, per comptar el nombre de vegades que l'usuari visita la pàgina, pots fer:

```php
<?php
    // Iniciamos la sesión o recuperamos la anterior sesión existente
    session_start();
    // Comprobamos si la variable ya existe
    if (isset($_SESSION['visits']))
        $_SESSION['visits']++;
    else
        $_SESSION['visits'] = 0;
?>
```

Si en lloc de el nombre de visites, voldries emmagatzemar l'instant en què es produeix cadascuna, la variable de sessió "visites" ha de ser una matriu i per tant hauràs de canviar el codi anterior per:

```php
// Iniciamos la sesión o recuperamos la anterior sesión existente
 session_start();
 // En cada visita añadimos un valor al array "visits"
 $_SESSION['visits'][] = time();
```

Tancar sessió

Encara que com ja has vist, pots configurar PHP perquè elimine de forma automàtica les dades de una sessió passat cert temps, en ocasions pot ser necessari tancar-la de forma manual en un moment determinat. Per exemple, si utilitzes sessions per recordar la informació d'autenticació hauràs donar-li a l'usuari de la pàgina web la possibilitat de tancar la sessió quan ho crega convenient.

En PHP tens dues funcions per eliminar la informació emmagatzemada en la sessió:

- `session_unset`. Elimina les variables emmagatzemades a la sessió actual, però no elimina la informació de la sessió del dispositiu d'emmagatzematge usat.

- `session_destroy`. Elimina completament la informació de la sessió del dispositiu de emmagatzematge.

Amb les funcions anteriors eliminem la informació però encara ens quedaria eliminar alguna cosa més. Saps quina?

Exercici pràctic

ACTIVITAT 3. REGISTRES DE VISITES

Modifica la pàgina `index.php` emmagatzemant en la sessió d'usuari els instants de totes les seves últimes visites. Si és la seva primera visita, mostra un missatge de benvinguda. En cas contrari, mostra la data i hora de totes les seves visites anteriors. A més a més, afegeix un botó a la pàgina que permeta esborrar el registre de visites.

3.6 Gestió de la informació de la sessió

Anem a utilitzar la informació de la sessió per a millorar alguns aspectes de la gestió del formulari de reserves.

Errors de validació

La pàgina de processament del formulari `reservations-store.php` mostra els errors de validació però no el formulari, que es troba en `reservations-create.php`.

El que farem serà controlar si hi ha errors en el processament i, en cas d'haver-ne, redireccionar a la pàgina `reservations-create.php`, emmagatzemant prèviament en la clau `errors` de la sessió els errors detectats.

Serà `reservations-create.php` qui controlarà si hi ha errors en la variable de sessió i en eixe cas els mostrarà.

Redireccions

Una redirecció és una tècnica que permet fer que el servidor web responga amb una URL diferent a l'actual.
Per a aconseguir s'utilitza la funció `header()`.

```php
// Send to the browser a new url and send a HTTP redirect code (302)
header('Location: /index.php')

// this function terminate the current script
exit();
```

Exercici pràctic

ACTIVITAT 4. ERRORS DE VALIDACIÓ
Modifica les pàgines `reservations-create.php` i `reservations-store.php` per a gestionar els errors tal com s'indica en l'explicació anterior.
Una vegada s'ha obtés la informació dels errors en la variable de sessió caldrà eliminar-la.

Missatges entre pàgines

Una altra qüestió a resoldre és la de la recàrrega de pàgines que han processat un formulari. Per exemple, què passa si després d'afegir una reserva actualitzem la pàgina? Es torna a intentar crear una reserva.

Per a solucionar-ho podem combinar de nou variables de sessió i redireccions. Si la reserva s'ha realitzat correctament podem redireccionar a la pàgina `index.php` per exemple, i usar una variable de sessió `message` per a passar a `index.php` la confirmació de que s'ha creat correctament la reserva.

Exercici pràctic

ACTIVITAT 5. MISSATGES ENTRE PÀGINES
Modifica les pàgines `reservations-store.php` i `index.php` per a gestionar els missatges tal com s'indica en l'explicació anterior.
Una vegada s'ha obtés la informació dels missatges en la variable de sessió caldrà eliminar-la.

Més informació sobre les sessions en PHP: Maneig de sessions (PHP)

3.7 Millores en el *framework* MVC

En el projecte `Movies` implementarem la classe `FlashMessage` que s'encarregarà de de gestionar tant els errors de validació com els missatges entre pàgines.