Tecnologies de Desenvolupament per a Internet i Web

Curs 2024-2025

Pràctica: *Botiga virtual.* Sessió 3

Índex

1 Dates

Grups	**Dia**
Grups A, B, C	04/11
Grup D	05/11
Grups E, F, G, H	06/11
Grups I, J, K	07/11
Grups L, M, N	08/11

2 Objectius

Pes	**Obligatori?**	**Funcionalitat**
Sistema de productes		
0'5	✓	Llistat de productes amb FETCH
0'4	✓	Detall de producte amb FETCH
Sistema d'usuaris		
0'6	✓	Registre: consultes parametritzades
0'3	✓	Registre: desar contrasenya xifrada
0'2	✓	El meu compte: menú amb `jQuery`

3 Feina prèvia abans de la sessió

- Mireu un tutorial de Javascript bàsic[8][5].
- Mireu un tutorial de FETCH[4].
- Mireu com executar consultes SQL parametritzades amb PHP i PostgreSQL[1][2].

4 Feina durant la sessió i abans de la següent sessió

4.1 Llistat de productes d'una categoria amb FETCH

Heu de modificar la pàgina del llistat de categories de manera que, en fer clic al títol —o la imatge, si és que en teniu— de cadascuna d'ella, es mostrin la informació resumida dels seus productes sense necessitat de recarregar la pàgina. Per implementar aquesta funcionalitat heu de fer **crides FETCH**.

La informació dels productes que heu de mostrar a aquesta pàgina és la següent:

- Títol.
- Imatge.
- Preu.
- Opcionalment, un botó d'afegir al cabàs.

A la figura 1 teniu un exemple del llistat de productes d'un dels webs d'un curs anterior.

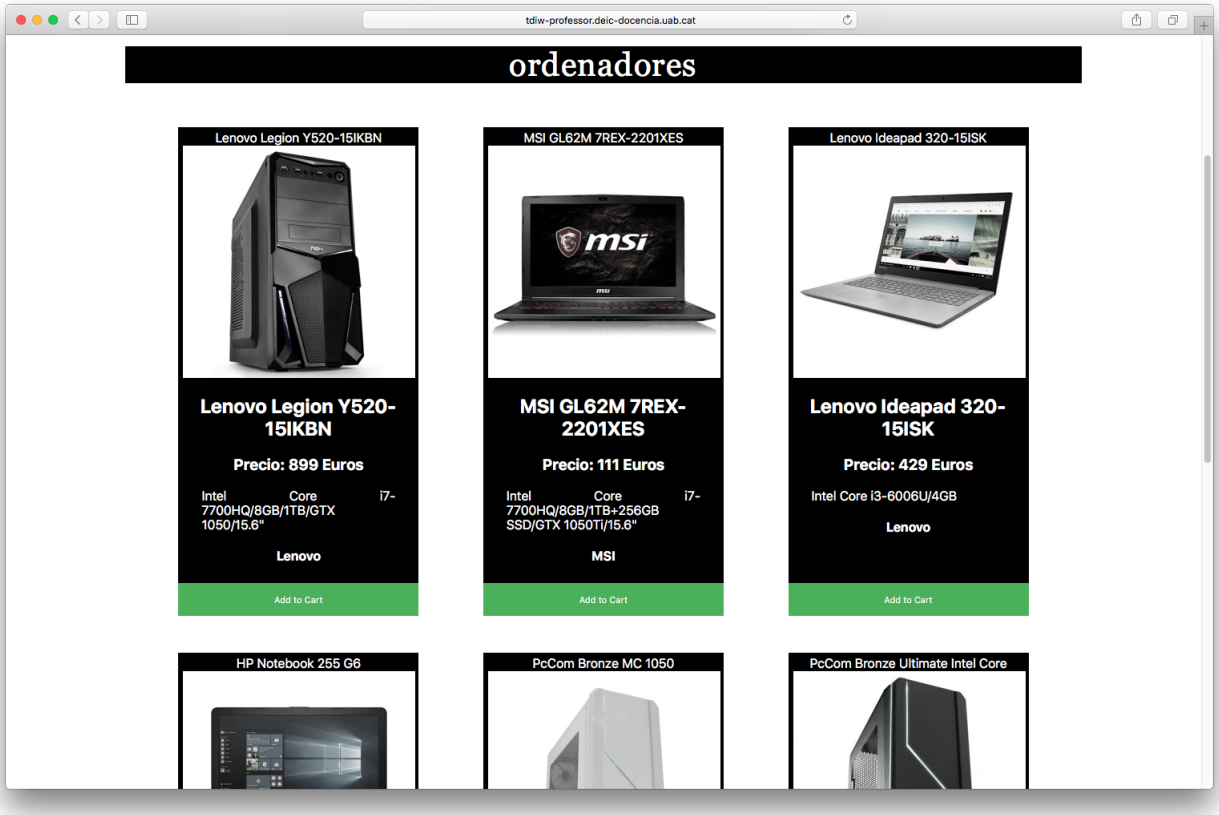

Figura 1: Exemple del llistat de productes d'un dels webs d'un curs anterior

4.2 Detall de producte amb FETCH

Igual que en el punt anterior, cal que mostreu la pàgina del detall d'un producte —la informació estesa— en fer clic a un producte des del seu llistat, emprant crides FETCH, és a dir, sense recarregar la pàgina.

La informació dels productes que heu de mostrar al seu detall és la següent:

- Títol.
- Descripció.
- Imatge (de moment no la teniu, però ho podeu anar preparant).
- Preu.
- Botó d'afegir al cabàs amb una quantitat, que encara no serà funcional.

☞ **Nota d'implementació:** En PHP, per a fer una consulta que té com a resultat un únic registre, si utilitzeu la funció **`pg_fetch_all`** haureu d'utilitzar la funció **`forEach`** per recórrer l'array resultat del **`fetch`** que conté només un registre de la base de dades. En aquest cas és millor que utilitzeu la funció **`pg_fetch_assoc`** [10], que us retorna directament un únic registre de la base de dades.

A la figura 2 teniu un exemple del detall de producte d'un dels webs d'un curs anterior.

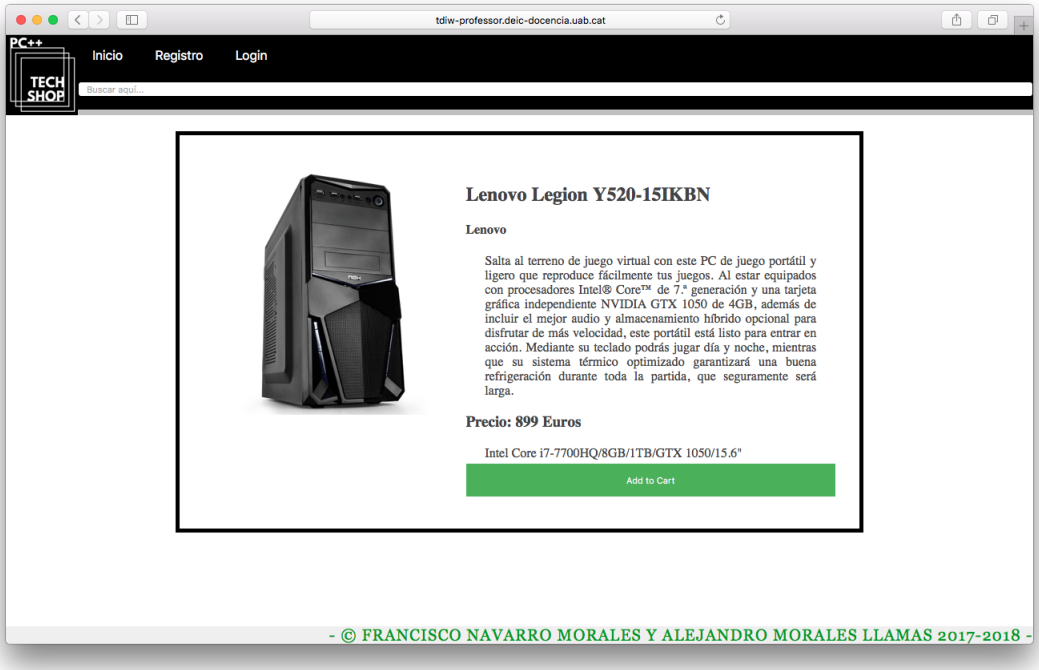

Figura 2: Exemple del detall de producte d'un dels webs d'un curs anterior

4.3 Registre d'usuari

A partir del formulari de registre d'usuari que vau fer a la sessió 1, heu d'emmagatzemar les dades de l'usuari a la base de dades un cop s'enviï el formulari. De moment no us heu de preocupar de filtrar les dades ni a la part del client ni a la part del servidor, però sí que heu de tenir en compte dos aspectes de seguretat de vital importància:

- Heu de xifrar la contrasenya. A PHP és molt senzill, heu de fer servir la funció `password_hash`[11], que és criptogràficament segura. **Les contrasenyes sempre s'han de desar xifrades a la base de dades. Tingueu en compte que el camp password a la base de dades ha de ser prou llarg (100chars) per poder guardar la contrasenya xifradda**.

- Heu de fer servir consultes parametritzades per emmagatzemar les dades de l'usuari. A la pràctica, per motius de planificació, només us demanem fer aquesta funcionalitat amb consultes parametritzades, però fóra bo que canviéssiu totes les consultes que ja teniu perquè s'executin com a consultes parametritzades, i que les noves consultes que feu a partir d'aquest moment les feu així. **Utilitzar consultes parametritzades permet evitar atacs d'injecció SQL**.

Tingueu present que, encara que feu el registre d'usuaris, de moment no podem iniciar sessió, això ho veurem a la següent sessió de pràctiques.

☞ **Nota d'implementació (opcional)**: Si vulguéssiu implementar el registre d'usuari amb una crida en AJAX, per invocar l'escript de registre amb un POST i passar-li els camps del formulari, vegeu la secció B de l'Apèndix.

4.4 Menú desplegable d'usuari

Heu d'afegir al vostre menú de navegació un menú desplegable amb les opcions corresponents a la navegació de l'usuari; aquest menú l'heu d'implementar amb jQuery. Penseu que aquest menú encara no serà totalment funcional, i que variarà en funció de si l'usuari ha iniciat sessió al seu compte o no.

Si l'usuari no ha iniciat sessió, l'única opció disponible al menú serà la d'iniciar sessió.

En canvi, si l'usuari ha iniciat sessió, les opcions d'aquest menú seran les següents:

- El meu compte.

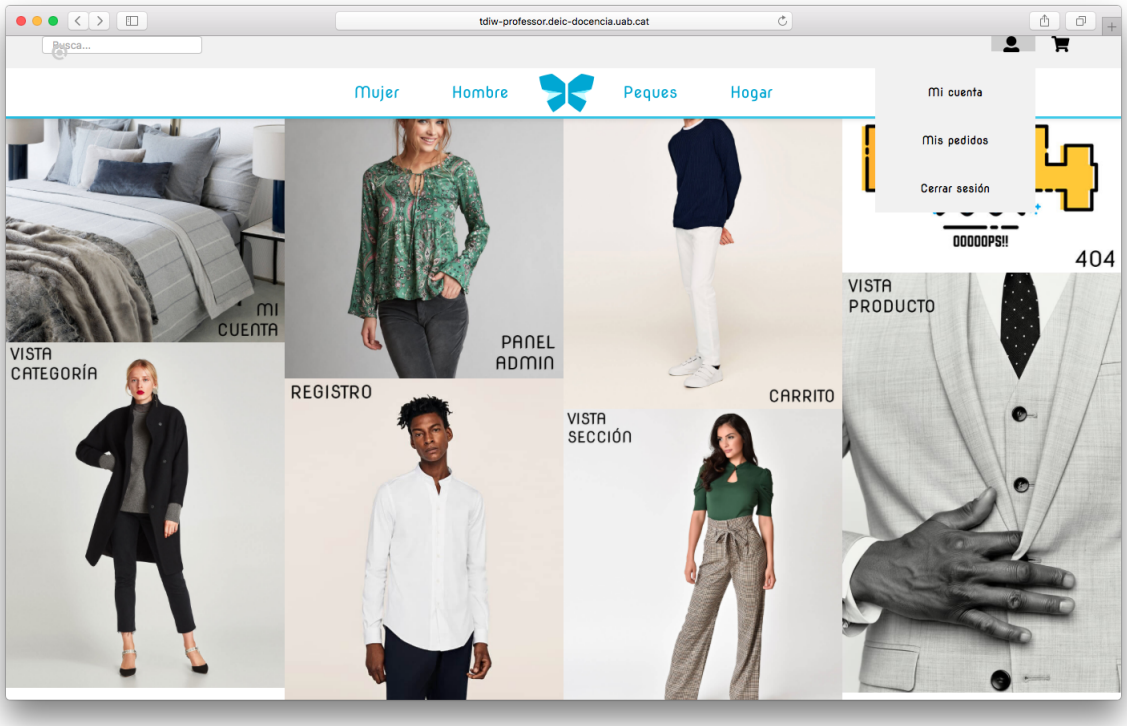

Figura 3: Exemple del menú desplegable d'un dels webs d'un curs anterior.

- Les meves compres.
- Tancar sessió.

A la figura 3 teniu un exemple del menú desplegable d'un dels webs d'un curs anterior.

Apèndixs

A Consultes parametritzades

Els atacs d'injecció SQL són els més comuns a aplicacions web[9]. És una tècnica que pretén obtenir el control de la base de dades a partir d'una consulta de l'aplicació. Sovint, aquests atacs exposen dades privades dels usuaris.

Prevenir-los és senzill, i per aconseguir-ho s'utilitza el que s'anomenen consultes parametritzades[1], que consisteixen a utilitzar un espai reservat o *placeholder* a les consultes. A aquest espai després s'hi afegeix el valor a consultar de manera segura.

Afortunadament, PHP ens permet executar consultes parametritzades de manera simple i senzilla amb el següent mètode:

A.1 Consultes parametritzades amb array de valors

Consisteix a afegir paràmetres amb un nom concret dins de la consulta SQL.

Suposem que volem agafar tots els productes d'una categoria, la que té l'identificador `3`. El codi PHP que tindríem per definir la consulta SQL podria ser com el següent:

```
1 <?php
2
3 $sql = 'SELECT *
4         FROM product
5         WHERE category_id = $1';
```

Hem preparat la consulta anterior per tenir-hi un paràmetre, `§1`, el qual ocuparà la primera posició (índex 0) del array de paràmetres que farem servir. El que ens cal, doncs, és assignar el valor d'aquest paràmetre a l'hora de fer la consulta.

El que s'ha de fer és simplement passar un *array* al mètode `pg_query_params`.

A aquest *array*, a la posició 0, hi hem de posar el valor que volem pel paràmetre *$1*:

[1] Heu de tenir present que una consulta parametritzada no consisteix en executar la consulta a la base de dades dins d'una funció que rep paràmetres.

```php
<?php

// Assignem manualment el valor de la variable per l'exemple
$categoryId = 3;

$sql = 'SELECT *
        FROM product
        WHERE category_id = $1';

$params = [$categoryId];

$result = pg_query_params($dbconn, $sql, $params);
$products = pg_fetch_all($result);
```

En executar la consulta, se substituirà `$1` pel valor de la variable `$categoryId`, que en aquest cas és `3`.

B Fer una crida AJAX per enviar dades al servidor utilitzant el mètode POST

Amb AJAX, per enviar dades per POST al servidor, o bé utilitzeu la funció **`fetch`** de JavaScript [3] o bé ho feu amb la llibreria **`JQuery`** [7] [6]. A continuació us mostrem tres exemples d'ambós opcions:

1. Utilitzant la funció **`fetch`** de javascript:

   ```javascript
   const response = await fetch("test.php", {
     method: "POST",
     body: JSON.stringify({ name: "John",
       passwd: "myPasswd",
       email: "John@example.org",
     }),
   });
   const data = await response.text();
   ```

   En l'exemple anterior, noteu com hem creat un diccionari amb els paràmetres que volem passar-li al servidor.

2. Utilitzant el mètode **`$.post`** de la llibreria **`JQuery`** [7]:

```
$.post( "test.php",
  { name: "John", passwd: "myPasswd", email: "John@example.org" },
  function(data, status){
    alert("Data: " + data + "\nStatus: " + status);
});
```

3. Utilitzant el mètode **$.ajax** de la llibreria **JQuery** [6]:

```
$.ajax({
  url : "test.php",
  type: "POST",
  data : { name: "John", passwd: "myPasswd", email: "Johns@example.org" },
  success: function(data)
  {
    //data - response from server
  },
  error: function (xhr,status,error)
  {
  }
});
```

Referències

[1] https://www.php.net/manual/en/function.pg-connect.php.

[2] https://www.php.net/manual/en/function.pg-query-params.php.

[3] developer.mozilla.org. Using the Fetch API. https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch.

[4] Mozilla Developers. Using the Fetch API. https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch.

[5] Javier Eguiluz. Introducción a javascript. http://www.librosweb.es/javascript/.

[6] JQuery.com. JQuery.ajax(). https://api.jquery.com/jQuery.ajax/.

[7] JQuery.com. JQuery.post(). https://api.jquery.com/jQuery.post/.

[8] MDN. Tutorials | MDN. https://developer.mozilla.org/ca/docs/Web/Tutorials.

[9] OWASP. Top 10-2017 Top 10. https://www.owasp.org/index.php/Top_10-2017_Top_10.

[10] php.net. Fetch a row as an associative array. https://www.php.net/manual/en/function.pg-fetch-assoc.php.

[11] php.net. How to calculate the hash of a password using a random salt. http://php.net/manual/en/function.password-hash.php.