Tecnologies de Desenvolupament per a Internet i Web
Curs 2016-2017

Pràctica: *Botiga virtual.*

Objectiu

Els objectius d'aquesta pràctica son aprendre les tecnologies bàsiques pel desenvolupament d'aplicacions web amb contingut dinàmic: `HTML5`, `CSS`, `Javascript`, `PHP+SQL`, `jQuery`, `JSON` i `Ajax`. Ser conscients del atacs als que s'exposen les aplicacions web i com evitar-los. Veure com implementar una aplicació web seguint un esquema Model View Controller. I, finalment, com podem programar la nostra aplicació seguint una arquitectura `REST`.
Per a fer-ho implementarem la nostra pròpia botiga virtual escollint la temàtica de la botiga de forma lliure. Cal, però, que **sigueu ètics**.
Us adjuntem un apèndix amb una explicació per a cada una de les tecnologies, llenguatges de programació i aspectes de seguretat que treballarem amb la pràctica.
A continuació us passem a detallar la funcionalitat que cal que implementi la vostra botiga. És obligatori que implementeu el que us demanem. No cal que us limiteu a les funcionalitats que us plantegem, podeu ampliar la botiga afegint tota la funcionalitat que us ajudi a ampliar el vostre coneixement en les tecnologies web.

1 Funcionalitat

L'aplicació web ha de tenir dues parts diferenciades:

Front-end La part amb la que interactuarà l'usuari.

Back-end La part interna, d'administració, de la botiga, amb un control d'accés per login i password.

A continuació us llistem els requisits funcionals de l'aplicació tant per la part del `front-end` com la del `back-end`.

1.1 Model View Controller, MVC

Tant pel que fa al *front-end* com al *back-end* els programarem seguint una arquitectura MVC orientada a web.

1.2 Control d'accés de la botiga virtual

Tal i com us expliquem en la secció 4 veureu que els fitxers font de la botiga virtual els desareu en el directori `public_html` del vostre compte de pràctiques. Protegirem l'accés a aquest directori i als seus subdirectoris mitjançant un *login* i un *password*. D'aquesta manera la resta de companys no podran accedir a la vostra botiga virtual ni descarregar-se els vostres recursos com els fitxers d'estil de *javascript*, etc...
Aquest control d'accés l'implementarem mitjançant la utilització dels fitxers *.htaccess* i *.htpasswd*. Vegeu com fer-ho a la secció 8.10 del l'apèndix.

1.3 Validació del codi HTML

Cal que el vostre codi `HTML5` segueixi els estàndards del llenguatge. Això s'aplica tant al front-end com al back-end.
Per a validar si el codi `HTML5` és correcte farem servir el verificador proporcionat pel *World Wide Web Consortium*. Trobareu l'enllaç al validador a [20].
El codi `HTML5` ha de passar obligatòriament el `w3validator`. Idealment fora bó que no tinguéssiu cap error, tot i que admetrem fins a 5 errors.

1.4 Validació dels fulls d'estil

El codi `CSS3` també haurà d'estar ben format. Això s'aplica tant al front-end com al back-end.
Per a validar els fulls d'estil farem servir el verificador proporcionat pel *World Wide Web Consortium*. Trobareu l'enllaç al validador a [18]. Idealment fora bó que no tinguéssiu cap error, tot i que admetrem fins a 5 errors.

1.5 URIs REST

Al principi del desenvolupament del projecte podeu obviar aquesta funcionalitat. Us explicarem com implementar-la durant la cinquena sessió de pràctiques.

No és obligatori però un recomanem **molt** que programeu la vostra aplicació per a que utilitzi URIs, peticions, en format `REST`.

El format de les peticions ha de ser el següent:
`http:/deic-dc0/~tdiw-xx/`*`recurs/acció/paràmetres`*
On:

recurs: L'utilitzareu per indicar el recurs sobre el que voleu fer una acció. Per exemple, podeu considerar els recursos: catàleg, producte, carret, compra, etc...

acció: L'utilitzareu per indicar l'acció que voleu fer sobre el recurs que ja heu especificat. Podeu considerar els següents possibles valors:

create: Donar d'alta.

read: Mostrar les dades d'un element.

update: Modificar.

delete: Eliminar.

paràmetres: L'utilitzareu per indicar quins són els paràmetres que requereix l'acció que ja heu especificat.

Per exemple, per veure els detalls del producte amb `id=12`, en lloc d'utilitzar la URI:

- **http:/deic-dc0/~tdiw-xx/getProd.php?id=12**

Utilitzeu:

- **http:/deic-dc0/~tdiw-xx/producte/read/12**

Per tal d'implementar aquesta funcionalitat us caldran dues passes:

1. Modifiqueu el fitxer `.htaccess` per dirigir les peticions REST que us arriben al servidor cap a un *wrapper*, l'escript `index.php`, que us les traduirà a crides als vostres escripts en `php`.
2. Convertiu el vostre escript `index.php` en un *wrapper* per fer les conversions de les peticions `REST` a les crides als vostres scripts.

1.6 Seguretat

Quan programeu una aplicació web heu de tenir molt present la seguretat de l'aplicació. El vostre website ha d'estar preparat per evitar els atacs més comuns de *Cross-site scripting* (XXS) i *Sql Injection* (SI). Vegeu més informació sobre aquests atacs i com evitar-los en les seccions de l'apèndix 9.1i 9.2.

També heu de tenir en compte que tota la informació sensible de l'aplicació que deseu a la base de dades cal que estigui xifrada. Vegeu com fer-ho a la secció 9.3.

1.7 Front end

El front end consisteix en la funcionalitat que directament veu l'usuari de la botiga virtual. A continuació us indiquem quines funcionalitats heu d'implementar.

1.8 Llistat de categories

Cal mostrar una llista de categories de productes. Aquesta llista pot ser d'un nivell o podeu considerar, si us calen, subcategories.
El llistat de categories és informació dinàmica que agafarem de la base de dades.

1.9 Llistat de productes

Per cada categoria cal mostrar el llistat dels productes associats.
Aquest llistat ha de mostrar:

- El nom del producte.
- Una imatge petita del producte.
- Una descripció curta del producte.
- El preu del producte.

El llistat de productes és informació dinàmica que agafarem de la base de dades.

1.10 Detall d'un producte

Quan seleccionem un producte hem de mostrar:

- La imatge del producte.
- La categoria a la que pertany.
- El nom.
- Un preu.
- Una descripció complerta del producte.
- Un enllaç o botó per afegir el producte al carret.

Quan afegim un producte al carret no abandonem la pàgina actual ni la tornem a carregar de nou. Indiquem de forma visual, amb un comptador que acaba d'incrementar-se en una unitat, que hi ha un producte més al carret. **Us caldrà utilitzar `AJAX`** per a implementar aquesta funcionalitat.

1.11 Cerca de productes

Cal que desenvolupeu una funcionalitat per poder cercar productes. Els productes s'han de poder buscar com a mínim pel seu nom.
Cal poder fer cerques a partir d'un nom aproximat. És a dir si faig una cerca amb la paraula: *php*, haig de poder obtenir resultats com:
PHP desde cero, Desarrollo Web Con PHP i MYSQL, Modern PHP, etc...

1.12 Carret de la compra

El carret de la compra haurà de poder emmagatzemar els productes que el client vagi seleccionant.
El contingut del carret s'ha de conservar durant tota la sessió de navegació de l'usuari.
El carret ha de ser visible, amb informació actualitzada, des de totes les pàgines del website.
La pàgina del carret ha de mostar:

1. Tots els productes seleccionats i el preu de cadascun.
2. La possibilitat d'eliminar un producte.
3. La suma total del preu dels productes que hi ha al carret.
4. Un botó o enllaç al formulari de compra.
5. Un botó o enllaç per tornar al catàleg de productes.

Un cop feta la compra, cal buidar el carret.

En el nostre cas **el carret de compra serà volàtil**. Si ens caduca la sessió de navegació o utilitzem un altre navegador o canviem d'ordinador no es mantindran les dades del carret de la compra.

1.13 Procés de compra

L'aplicació haurà de permetre realitzar la compra dels productes que conté el carret. Per a que un usuari pugui efectuar una compra cal que, prèviament, estigui registrat.
El registre consisteix en mostrar a l'usuari un formulari amb els camps que trobareu especificats en la secció 1.14.
El procés de compra seguiria el següent diagrama de flux:

1. Des del carret de la compra, quan l'usuari selecciona "comprar" li apareix un formulari de *login* i un botó o enllaç a una pàgina de registre.

2. Si l'usuari ja s'havia registrat prèviament:

   (a) L'usuari ha d'entrar el seu usuari i password.

   (b) Anar al pas 4.

3. Si l'usuari no s'havia registrat prèviament:

   (a) L'usuari ha de seleccionar el botó o enllaç a la pàgina de registre.

   (b) Es carrega un formulari de registre que l'usuari ha d'omplir. Aquestes dades s'han de guardar a base de dades per poder-les utilitzar en totes les compres que faci l'usuari.

   (c) Un cop omplert i validat el formulari de registre anar al pas 4. Per validar el formulari llegiu la secció 1.14.

4. A l'usuari se li mostra una pàgina per a que confirmi la compra que està apunt de dur a terme. En aquesta pàgina de confirmació heu de mostrar el preu total de la compra i un parell de botons, un per tirar endavant la compra i un altre per cancel.lar-la.

   Si ho creieu convenient, en aquesta pàgina li podeu tornar a mostrar a l'usuari el nom dels productes que està apunt de comprar. També podeu ampliar la pràctica donant-li a l'usuari la possibilitat de canviar l'adreça d'enviament. Aquestes funcionalitats no puntuaran.

5. Un cop feta la compra a l'usuari se li mostra un missatge d'agraïment i un enllaç o botó per tornar al catàleg de productes.

6. Un cop feta la compra es buida el carret.

1.14 Validació de les dades del formulari de registre

S'ha de comprovar que tots els camps del formulari són correctes. Com a mínim heu de considerar els següents camps i les següents validacions.

Camp	Obligatori	Regles de validació
Nom	✓	Només caràcters i espais.
Usuari	✓	Camp alfanumèric. No poden haver-hi espais.
Password	✓	Camp alfanumèric.
Adreça de correu	✓	Ha de contenir un correu vàlid que contingui els caràcters: @ i '.'
Telèfon	✓	Ha de contenir 9 dígits.
Adreça	✓	Pot contenir fins a 30 caràcters alfanumèrics.
Població	✓	Pot contenir fins a 30 caràcters i espais.
Codi Postal	✓	Ha de contenir 4 dígits.
Targeta bancària	✓	Ha de contenir 16 dígits

La validació la farem en la banda del **client** i en la banda del servidor. A continuació us indiquem com fer-ho en cada cas.

1.14.1 Validació en client

La validació en la banda del client cal fer-la per a avisar immediatament a l'usuari dels possibles errors que hagi comés en l'entrada de dades en un formulari abans d'enviar aquestes dades al servidor.
Aquesta validació la farem utilitzant els atributs del tag input de HTML5. Vegeu els atributs disponibles a [31].

També podeu utilitzar `javascript` per fer la validació.

Un atacant fàcilment es pot saltar la validació de les dades que es fan en el client per a injectar dades no validades al servidor. És per aquest motiu que també heu de tenir la precaució de validar les dades en el servidor abans d'utilitzar-les. A continuació us indiquem com ho heu de fer.

1.14.2 Validació en el servidor

La validació en el servidor constarà de dues passes:

1. En primer lloc farem un sanejament de les dades que ens venen del formulari. Vegeu com es fa a [29].

2. En segon lloc farem la validació del format d'aquestes dades. Aquesta validació la podeu fer utilitzant les funcions PHP `filter_var` [5] o bé `preg_match` [7].

   La funció `filter_var` consisteix en aplicar uns filtres de format a les dades que volem validar. Aquests filtres ja estan predefinits.

   Mitjançant aquesta funció també podem fer el sanejament de les dades. Així doncs podeu triar de quina manera voleu fer el sanejament de les dades.

   Vegeu exemples d'utilització de la funció `filter_var` en [6].

   Pel que fa a la utilització de la funció `preg_match` consisteix en comprovar si les dades satisfan una expressió regular que heu de passar com a paràmetre.

   Podeu utilitzar les funcions que més s'adaptin a les vostres necessitats.

1.15 Menú desplegable amb les funcionalitats associades al compte d'un usuari: My Account

Cal que desenvolupeu un menú desplegable per poder accedir a les diferents funcionalitats associades al compte d'un usuari registrat.
Aquest menú és accessible a través d'un botó o enllaç que faci referència al compte d'usuari, `My Account`.
Si l'usuari encara no s'ha identificat, en accedir a l'enllaç de `My Account`, se li ha de mostrar el formulari per, o bé identificar-se, o bé registrar-se.
En cas que l'usuari s'hagi identificat, en clicar a l'enllaç de `My Account` s'han de desplegar les operacions de: "Les meves dades", "Les meves compres", altres opcions que creieu convenients. Aquest menú desplegable l'**heu de desenvolupar en `JQuery`**.

En cas que, pel vostre disseny de la pràctica, ja hagueu utilitzat un menú desplegable en `JQuery`, la funcionalitat del compte d'usuari la podeu desenvolupar amb la tecnologia que millor us vagi.
A continuació us indiquem la funcionalitat que heu de desenvolupar en les dues opcions del menú de `My Account`:

Les meves dades En aquesta opció de menú heu de mostrar un formulari amb les dades de l'usuari que heu especificat en el procés de registre. L'usuari ha de poder modificar qualsevol camp del formulari. En aquest cas s'ha de tornar a validar el formulari i s'han de modificar les dades a la base de dades.

Les meves compres En aquesta opció heu de llistar l'històric de les compres de l'usuari. Per a cada compra heu de mostrar la data en la que es va dur a terme i els productes, amb el seu preu, que es van comprar.

1.16 Back End

El back end de la botiga virtual implementa les funcionalitats d'administració de la botiga.
Heu d'oferir un menú per poder gestionar productes i compres.
En les següents seccions us especifiquem com heu de desenvolupar aquests funcionalitats.

1.17 Control d'accés de la part d'administració

La part d'administració ha d'estar protegida amb un formulari on es demani el `login` i `password` per a que cap usuari, tret d'un administrador, pugui accedir-hi. **Per a tots els grups** el login i el password serà **admin**.
Com en el cas de l'alta d'un usuari, aquest password s'ha de desar en format hash en la base de dades.

1.18 Gestió de productes: CRUD (Create, Read, Update and Delete)

En el back-end, quan cliqueu l'opció de gestionar productes heu de mostrar un formulari que contingui els següents elements:

- Un botó o enllaç per donar d'alta un nou producte. Per a més detalls vegeu la secció 1.19.
- Una llista desplegable que contingui totes les categories. Aquesta llista s'ha de crear de forma dinàmica amb la informació que tenim a la base de dades. A través de cada categoria, mitjançant d'un enllaç, botó, o algun altre mecanisme, heu mostrar-ne els productes associats.

  En cas que considereu subcategories, quan seleccioneu una categoria, heu d'omplir una altra llista desplegable amb les subcategories de la categoria seleccionada. **Aquesta funcionalitat l'heu de desenvolupar amb `Ajax`**.

Mitjançant **`Ajax`**, quan cliqueu el botó per llistar els productes, heu de carregar la llista de productes de la categoria just a sota de la llista desplegable de categories. La idea és que no torneu a generar de nou la llista de categories.
Al costat de cada producte heu de tenir un enllaç o botó o oferir d'alguna manera la possibilitat d'eliminar el producte o bé de poder-lo editar per modificar-lo.

1.19 Donar d'alta un nou producte

En el formulari per donar d'alta nous productes com a mínim s'han de poder especificar els següents camps:

- Títol del producte.
- Llista desplegable amb les categories i, si cal, llista desplegable amb les subcategories. Aquestes dades son dinàmiques i s'han d'obtenir de la base de dades.
- Descripció curta.
- Descripció llarga.
- Preu.
- Camp per poder pujar la imatge del producte al servidor.

  Aquesta funcionalitat la desenvoluparem amb `PHP`. Vegeu les passes a seguir en [4].

1.20 Gestió de les compres

Per a cada compra cal que mostreu: La data en la que es va fer la compra, el nom complert del client, la llista de productes inclosos en la compra, indicant-ne el seu preu, i l'import final de la compra.

2 Consideracions de sentit comú

A continuació us recordem quines consideracions heu de tenir en compte. Aquestes consideracions són de sentit comú però no està de mes recordar-vos-les.

2.1 Estructura correcta

Una estructura correcta ha de complir els següents aspectes:

2.1.1 Navegació fàcil

Cal que la web sigui usable, fàcil de fer servir:

- Cal que les seccions estiguin ben diferenciades i identificades.
- Cal estalviar el màxim número de clics a l'usuari.
- Cal que el procés de compra sigui senzill.

2.1.2 Estils i estructura homogenis

Cal que totes les seccions del web segueixin el mateix estil i estructura, que no hi hagi pàgines redundants, i que s'utilitzin els mecanismes d'inclusió adients **evitant la repetició de codi**.
Sempre que pugueu identifiqueu les parts comunes a totes les pàgines del *website* i guardeu-les en un fitxer. Així només us caldrà incloure aquest fitxer a totes les pàgines que el necessitin. Això ho farem amb PHP.

2.2 Idoneïtat de la base de dades

La base de dades haurà de tenir una estructura adient.

3 Planificació de la pràctica

A continuació us indiquem què treballarem durant cada sessió de pràctiques.
L'organització de les sessions de pràctiques sempre serà la mateixa, excepte la sessió on es fa una entrega parcial i d'entrega final.
Durant la sessió, al laboratori, treballareu sobre una funcionalitat de la pràctica i demanareu tots els dubtes que us sorgeixin.

Per a cada sessió cal que feu un treball previ indispensable per poder aprofitar la sessió de la pràctica. Aquest treball previ normalment consistirà en que us mireu com funciona alguna tecnologia o paradigma de programació que ens caldrà per desenvolupar la funcionalitat de la pràctica que toqui. La idea és que no gastem temps en mirar tots junts un tutorial sobre una tecnologia sinó que us ho mireu prèviament a casa. Durant la sessió heu de preguntar els dubtes que us hagin sorgit sobre aquesta tecnologia que us heu mirat a casa i heu de desenvolupar la funcionalitat que us demanem.

3.1 Sessió 1

Tots els grups: setmana del 26/09.

Feina prèvia que heu de fer abans de venir a la sessió

- Repasseu un tutorial de HTML5 [23].
- Repasseu un tutorial de CSS [17].
- Penseu què vendreu en la vostra botiga.

- Dissenyeu el prototipus de com serà la botiga. Per a fer aquest disseny podeu utilitzar una eina gràfica com per exemple: `http://moqups.com`. Aquesta web us permetrà fer, *online*, dissenys i desar-los a disc. A més està feta íntegrament en `HTML5` amb el que no us caldrà l'ús de cap *plugin*.
- Repasseu la secció 1.2 i l'apendix 8.10.

Feina que heu de fer durant la sessió i acabar a casa abans de la sessió següent

3.1.1 Control d'accés al directori del vostre website

Configureu els fitxer `.htaccess` i `.htpasswd` per a protegir amb password l'accés a la vostra botiga.

3.1.2 Pàgina del catàleg

Desenvolupeu el layout, l'estructura, en **`HTML5`**, de la botiga. Comenceu a desenvolupar les pàgines del vostre *website*. Implementeu la pàgina on es mostren les categories i el llistat de productes d'una categoria i els detalls d'un producte. De moment aquesta informació serà estàtica.

3.1.3 Formulari de registre d'un usuari

Desenvolupeu el formulari de registre d'un usuari. Vegeu la secció 1.14. No cal que afegiu el formulari a la navegació de la web. Ja ho farem més endavant.
En els camps del formulari utilitzeu els atributs HTML5 que us calguin per a forçar a que l'usuari entri les dades de forma correcta. Estem validant els camps del formulari en el client abans d'enviar-los al servidor.

3.1.4 Desenvolupeu la pàgina de login de l'usuari i de l'administrador

En el formulari de login l'usuari especifica el seu login i password. Cal que afegiu aquests dos formularis a la navegació de la botiga.
En els camps del formulari utilitzeu els atributs HTML5 que us calguin per a forçar a que l'usuari entri les dades de forma correcta

3.1.5 Estils

Definiu els estils, en **`CSS3`**, de la botiga.

3.2 Sessió 2

Grups	Setmanes
Grups F i G	17/10
Resta de grups	10/10

Feina prèvia que heu de fer abans de venir a la sessió

- Mireu un tutorial de PHP bàsic [12].
- Mireu un tutorial de com crear objectes amb PHP [11].
- Mireu un tutorial com connectar-se a una base de dades MySQL mitjançant PHP **Orientat a Objecte** [13].
- Repasseu el paradigma de programació del `MVC`.
- Porteu en un fitxer, o en paper, el diseny de la vostra base de dades per a que el professor us el pugui validar.

Feina que heu de fer durant la sessió i acabar a casa abans de la sessió següent

3.2.1 MVC

Tot el website l'heu de programar seguint el paradigma de programació `MVC` per a aplicacions web. Per a tal efecte seguiu les següents passes:

1. Creeu els directoris: `model`, `view` i `controller`.
2. Modifiqueu el `index.html` renombrant-lo a `index.php` per a que faci un *require* del *controller* que ha de pintar la pàgina inicial.
3. Modifiqueu les pàgines que ja heu desenvolupat per a que segueixin el patró `MVC`.

3.2.2 Generació dinàmica de les categories i dels productes

Cal que mostreu les categories dels productes que llegiu de la base de dades. Vegeu la secció 1.8. Per a cada categoria cal que mostreu els productes associats que hi ha a la base de dades. Vegeu la secció 1.9.
Recordeu d'implementar aquesta funcionalitat seguint el patró del MVC.

Sessió 3

Grups	Setmanes
Grups F i G	31/10
Resta de grups	24/10

Feina prèvia que heu de fer abans de venir a la sessió

- Mireu un tutorial de `JavaScript` bàsic [24] [15].
- Mireu un tutorial de `jQuery` [25] [3].
- Mireu un tutorial de `AJAX` [21] [16].
- Mireu com funcionen els atacs de SQL Injection (SQLI) en 9.2 i vegeu com evitar-los.
- Mireu com funcionen les consultes SQL parametritzades amb PHP. Vegeu la secció 9.4 i consulteu [14].

Feina que heu de fer durant la sessió i acabar a casa abans de la sessió següent

3.2.3 Mostrar els productes associats a una categoria

Modifiqueu la pàgina on mostreu els productes d'una categoria per a que, **mitjançant `JavaScript` i `Ajax`**, quan seleccioneu una categoria es mostrin els productes associats a la categoria sense haver de carregar de nou la pàgina. Si utilitzeu subcategories utilitzeu aquesta tecnologia per a pintar la llista de les subcategories sense carregar la pàgina de nou.

3.2.4 Mostrar el menú del compte d'un usuari: *My Account*

El menú que heu de desenvolupar és l'associat al compte d'usuari. Vegeu tota la infomació a 1.15. Aquesta funcionalitat l'heu de desenvolupar en `jQuery`. Per ara no cal que l'usuari s'hagi identificat per poder accedir a aquest menú.

3.2.5 Guardar a la base de dades les dades de registre de l'usuari

Quan l'usuari faci un *submit* del formulari cal que deseu les dades a la base de dades. **Utilitzeu una consulta parametritzada per a fer-ho**. Les dades del formulari les validarem la sessió vinent. Procureu d'utilitzar consultes parametritzades per totes les consultes que hagueu de fer a la base de dades.

Sessió 4: Entrega parcial de la pràctica

Tots els grups: setmana del 14/11.

Entrega parcial de la pràctica

Durant aquesta sessió s'avaluarà que la vostra pràctica implementi les funcionalitats descrites en les sessions de la 1 a la 3.
Cal que deseu els fitxers en el servidor `deic-dc0.uab.cat`. Provarem els fitxers allotjats en aquest servidor.
Utilitzarem la versió del Chrome del laboratori per provar la funcionalitat de la pràctica.
També s'avaluarà de forma individual a cada un dels membres del grup de pràctiques. Si algún membre no demostra haver adquirit els coneixements necessaris li quedarà la pràctica suspesa.

Feina prèvia que heu de fer abans de venir a la sessió

- **Acabar tota la funcionalitat descrita en les seccions 1, 2 i 3**.
- El password de l'usuari l'heu de desar a la base de dades xifrat. No pot estar en clar. Mireu com fer-ho a la secció 9.3.
- Mireu-vos com manipul.lar cookies amb php. Vegeu [27].
- Mireu-vos com mantenir dades durant tota la sessió de navegació mitjançant PHP. Vegeu [28].

Feina que heu de fer durant la sessió i acabar a casa abans de la sessió següent

3.2.6 Xifrar el password de l'usuari i el de l'administrador

El password que l'usuari especificat en el formulari de registre s'ha de desar xifrat en la base de dades.

Sabeu que tindreu un usuari administrador de la botiga virtual. El login i el password serà respectivament `admin`, `admin`. Per desar el password xifrat a la base de dades el calcularem amb la comanda `htpasswd` amb la següent crida: `htpasswd -n admin`

Amb aquesta comanda genereu un password per l'usuari admin que es mostrarà per pantalla. La sortida seria similar a la següent:
admin:$apr1....

El password comença a partir del primer símbol $. Aquesta cadena l'heu de desar a la base de dades.

3.2.7 Carret de la compra

A continuació us indiquem la funcionalitat que heu d'implementar del carret de la compra:

1. Poder-hi afegir productes.
2. El contingut del carret s'ha de conservar durant tota la sessió de navegació de l'usuari.
3. El carret ha de ser visible des de totes les pàgines del website.
4. S'ha de visualitzar en tot moment el nombre de productes del carret.
5. Quan afegim un producte al carret automàticament s'ha de visualitzar l'increment del seu contingut. Aquesta funcionalitat l'haureu d'implementar en AJAX. Podeu triar si voleu utilitzar javaScript o jQuery per implementar aquesta petició AJAX.
6. Cal implementar la pàgina del carret on es mostren tots els productes seleccionats i la possibilitat d'eliminar-los.

Sessió 5

Tots els grups: setmana del 28/11.

Feina prèvia que heu de fer abans de venir a la sessió

- Mireu-vos com funcionen els atacs de Cross Site Scripting (XSS) en 9.1 i vegeu com evitar-los.
- Mireu-vos com validar els camps del forulari en el servidor. Vegeu la secció 1.14.
- Mireu-vos com pujar un fitxer al servidor a través d'un formulari HTML5. Vegeu com es fa en PHP a [4].
- Mireu-vos la secció 1.5.

Feina que heu de fer durant la sessió i acabar a casa abans de la sessió següent

3.2.8 Validació del formulari de registre d'un usuari

En la secció 1.14 teniu tota la informació per poder validar el formulari de registre d'un usuari.

3.2.9 Formulari d'alta d'un producte

Cal que tingueu el formulari del *backend* per donar d'alta productes. Mitjançant aquest formulari cal que pugueu pujar la imatge del producte al servidor.

3.2.10 URIS REST

Modificarem l'aplicació per a poder accedir-hi mitjançant URIs REST. Vegeu la secció 1.5.

Sessió 6

Tots els grups: setmana del 19/12.

Entrega i avaluació de la pràctica.
L'avaluació de la pràctica consistirà en provar l'aplicació i fer preguntes sobre el codi als diferents membres del grup de forma individual.
En cas de que l'alumne no sàpiga contestar o contesti malament li quedarà la pràctica suspesa.

4 Entorn de la pràctica

- Aquesta pràctica es farà utilitzant un servidor amb sistema operatiu Linux. El servidor web que utilitzarem és l'*Apache*, del qual es pot trobar més informació a `http://www.apache.org`. El servidor Apache s'està executant en la màquina `deic-dc0.uab.cat`.

- Cada grup de treball disposarà d'un compte de pràctiques, en el vostre compte tindreu creat el directori *public_html*. Tot el que hi hagi en aquest directori es veurà públicament a: **http://deic-dc0.uab.cat/~usuari**. Dins d'aquest directori heu de crear el fitxer *index.html*, que serà la pàgina inicial de l'aplicació. Aquesta pàgina es carregarà automàticament en accedir a la vostra adreça.

- Per accedir al vostre compte de pràctiques remotament ho heu de fer via **ssh**.

  Si esteu en un entorn Linux, directament heu d'executar:

  ```
  ssh <usuari>@deic-dc10.uab.cat
  o
  ssh <usuari>@deic-dc26.uab.cat
  ```

  Si esteu en un entorn Windows podeu utilitzar l'aplicació **putty**[1].

- El professor us proporcionarà un password temporal per accedir al vostre compte. Un cop hagueu entrat per primera vegada, canvieu-lo utilitzant la comanda `yppasswd`

- Per copiar fitxers remotament, si esteu en un entorn Linux, podeu utilitzar les comandes *scp i sftp*. En entorns Windows podeu utilitzar l'aplicació *winscp*[2].

- La pàgina web de pràctiques és: **http://deic-dc0.uab.cat/tdiw/practiques**. En aquesta pàgina hi trobareu els enllaços d'accés a les vostres webs.

- Al laboratori us hem instal.lat els editors `brackets` amb les extensions `PHP Syntax hint` i `W3CValidation` i l'editor `sublime`. Per executar-los, des de la línia de comandes executeu: `backets` i `subl` respectivament.

- El servidor `deic-dc0.uab.cat` treballa amb les següents versions del software:

Apache	2.4.10
MySQL	5.5.49
php	5.6.22

- Podeu simular l'entorn de la pràctica a casa vostra instal.lant-vos un servidor Apache amb módul de PHP i una base de dades MySQL. Els següents paquets contenen aquest software: Per a Linux `LAMP` (Linux Apache MySQL PHP), per a Mac `MAMP` i per a Windows `WAMP`.

5 Condicions de lliurament

- La web es corregirà utilitzant l'última versió disponible del navegador *Chrome*.
- **La web s'haurà d'allotjar al servidor deic-dc0.uab.cat**. L'avaluació del control i de l'entrega final es realitzarà amb la versió allotjada al servidor. Si no es compleix aquest requisit us quedarà un **zero** de l'avaluació corresponent.
- El **lliurament** final de l'aplicació es realitzarà a l'**última sessió**, fent servir la versió que hi hagi penjada al servidor **1 hora** abans de l'inici d'aquesta.

6 Nota de pràctiques

A continuació us indiquem com es calcularà la nota final de la pràctica.

- **Per poder tenir nota de la pràctica cal que l'assistència a les sessions de laboratori sigui major o igual al 80%**. Si l'assistència és menor quedarà un 0 de la pràctica.
- Tant l'entrega parcial com la final puntuaran sobre 10.
- L'entrega parcial serà un 30% de la nota final i l'entrega final serà un 70%.
- Si no se supera l'avaluació de l'entrega parcial o bé l'avaluació de l'entrega final, la nota de la pràctica serà un 0.

Així doncs la fórmula que per calcular la nota de la pràctica seria la següent:

```
(30%(nota_entrega_parcial) + 70%(nota_entrega_final) ) *
(apte_entrega_parcial) * (apte_entrega_final) * apte_assitència
```

A continuació us indiquem quina serà la rúbrica de la correcció del lliurament de la pràctica.

6.1 Rúbrica de la correcció del lliurament final

Tot seguit es mostra una taula amb la puntuació de cada funcionalitat que s'ha d'implementar. Les funcionalitats marcades com a obligatòries amb un *check* són imprescindibles per poder avaluar-vos la pràctica. Si us falta o falla alguna d'aquestes funcionalitats la nota de la pràctica serà 0.

Pes	✓	**Funcionalitat**
0.8	✓	MVC.
0.3	✓	Llistat de categories.
0.4	✓	Llistat de productes d'una categoria amb AJAX.
0.3	✓	Mostrar els detalls d'un producte amb AJAX.
0.25	✓	Identificació de l'usuari.
0.25		Registre d'un usuari: validació en client.
0.4		Registre d'un usuari: validació en servidor.
0.5	✓	Registre d'un usuari: desar les dades amb parametritzades.
0.4	✓	Registre d'un usuari: desar el password xifrat.
0.4	✓	Menu desplegable de My Account amb jQuery.
0.25		My Account: veure les dades de l'usuari.
0.3		My Account: Modificar les dades de l'usuari des d'un formulari ple amb les dades antigues.
0.3		Cercador de productes.
0.3	✓	Carret visible desde tot el website.
0.5		Carret: Actualitzar el nombre de productes amb AJAX.
0.5	✓	Carret: Contingut del carret.
0.25	✓	Compra: identificació del client.
0.2		Compra: Confirmació de la compra.
0.3	✓	Compra: desar la compra a la base de dades.
0.2		Compra: buidar el carret.
		BackEnd
0.3	✓	Control d'accés per a l'administrador des de tot el backend.
0.2	✓	Menú per gestionar productes i compres.
0.3	✓	Alta de producte amb imatge
0.2	✓	Accedir a un producte a través de les categories.
0.2		Carregar les dades del producte a modificar.
0.2		Modificar un producte.
0.2		Llista de les compres
0.8		Suportar URIs amb format REST
0.25	✓	Validació HTML5 ben format del formulari de registre.
0.25		Validació CSS ben format.
10		TOTAL

* Per aprovar el lliurament final cal que la nota sigui superior a 5 i que estiguin implementades les funcionalitats mínimes marcades amb el símbol ✓.

7 Recuperació

Aquells alumnes que no assoleixin els mínims per aprovar la pràctica, podran realitzar un segon lliurament de la mateixa pràctica a principis de febrer.
Els criteris d'avaluació que es faran servir seran els mateixos però **la nota màxima a la que es podrà optar serà un 8**.
Per aprovar cal que la nota sigui superior a 5 i que estiguin implementades les funcionalitats mínimes asenyalades amb un `check`. El lliurament s'haurà de realitzar presencialment i respondre les preguntes que pugui plantejar el professor.
Així doncs la nota final de recuperació es calcularà de la següent manera:

```
(nota_entrega_repesca) * (apte_funcionalitats_minimes) *
(apte_validacio)
```

8 Apèndix 1: Tecnologies web

Per a implementar la web utilitzarem les següents tecnologies:

8.1 HTML5

[19, 23] Llenguatge d'edició anomenat `HTML`, `HyperText Markup Language`. Aquest llenguatge, basat en marques d'estil, permet definir l'estructura bàsica i el contingut d'una pàgina web d'una manera estàndard i eficient. El navegador web interpreta aquest llenguatge.

8.2 CSS

[17, 22] Amb la creixent complexitat del llenguatge `HTML`, i donada l'heterogeneïtat de la informació, es va veure la necessitat de separar contingut o informació de la seva presentació (aspecte i format). D'aquesta necessitat va sorgir el CSS (Cascading Style Sheets), un llenguatge de fulls d'estil emprat per donar un estil particular a pàgines web escrites amb HTML. D'aquesta manera, mitjançant l'HTML especifiquem el què: un paràgraf, una imatge, mentre que amb els full d'estil CSS especifiquem el com: font i color del text, mida de l'imatge, etc... .

8.3 Javascrip

[24, 15] És un llenguatge que **s'executa en el client**, navegador. Ens permet donar dinamicitat a la web i que l'usuari hi pugui interaccionar.
Ens permet fer validacions dels formularis, fer crides asíncrones al servidor mitjançant `Ajax` i detectar events que fa l'usuari sobre la web.

8.4 JQuery

[25] És una llibreria de `Javascript` que simplifica la utilització de Javascript.

8.5 AJAX

[21, 16] És una tecnologia que fa servir `Javascript` i `XML` per a fer crides asíncrones al servidor. **S'executa en el client**, navegador.
Ens permet actualitzar una part la pàgina de forma dinàmica. D'aquesta manera el client pot demanar nova informació al servidor sense haver de recarregar tota la plana web.

8.6 JSON

JSON, `JavaScript Object Notation`, és un format de dades basat en text molt lleuger pensat per a l'intercanvi de dades estructurades.
Aquest format és "*human readable*" i és molt senzill de parsejar.
Forma part de l'estàndard del llenguatge de programació JavaScript. La sintaxis d'un missatge `JSON` és idèntica a la sintaxis per definir un objecte en JavaScript.
Degut a la seva senzillesa i lleugeresa, aquest format s'utilitza en qualsevol àmbit on hi han dades, estructurades, que s'han de transmetre.
L'estructura és molt similar a la de `XML`, només varia el format.
JSON ens permet representar dos tipus de dades estructurades: objectes i arrays.
Un objecte es defineix de la següent manera:

```
object
    {}
    { members }
members
    pair
    pair , members
pair
  string: value
value
    string
    number
    object
    array
    true
    false
    null
```

Un array es defineix com:

```
array
    []
    [ elements ]
elements
    value
    value , elements
value
    string
    number
```

```
object
array
true
false
null
```

En el següent exemple definim un objecte persona en JSON:

```
{"firstName":"John", "lastName":"Doe"}
```

En el següent exemple definim l'objecte treballadors d'una empresa en JSON:

```
{"employees":[
    {"firstName":"John", "lastName":"Doe"},
    {"firstName":"Anna", "lastName":"Smith"},
    {"firstName":"Peter","lastName":"Jones"}
]}
```

Vegeu més informació a [26].

8.7 PHP

[12] Moltes vegades el contingut d'una pàgina no és estàtic sinó que s'actualitza amb el temps. Per exemple, una botiga virtual que afegeix, elimina o modifica algun dels seus productes. Tot i que això es podria fer de forma manual, modificant el codi HTML per reflectir els canvis, no és recomanable, ni molt menys, eficient. Per generar webs de forma dinàmica comptem amb el llenguatge de scripting **PHP** (*PHP Hypertext Preprocessor*).
El codi PHP és interpretat pel servidor, el qual haurà de tenir un mòdul de processament PHP, per **generar el codi HTML de la pàgina web resultant**.
El codi PHP es pot "incrustar" directament en la pàgina HTML origen sense necessitat de fer servir fitxers addicionals.
Quan una pàgina HTML té codi php incrustat passa a tenir l'extesió `.php`, per exemple `index.php`.

8.8 PHP + Orientat a objecte

En `PHP` també es poden definir classes i, per tant, objectes o instàncies d'aquesta classe.
Una classe és una estructura de dades que pot contenir constants, variables i mètodes. A continuació us mostrem un exemple d'una classe que té una variable, una constant un mètode i un mètode estàtic.
Noteu que les constants són estàtiques, és a dir no cal crear un objecte de la classe per accedir-hi. Amb el modificador *static* també podeu definir mètodes estàtics.

```
<?php
class SimpleClass
{
  // contant declaration
    public const MAX = 100;

    // property declaration
    public $var = 300;

    // method declaration
    public function displayVar() {
        echo $this->var;
    }

    public static function foo(){
       echo "hello world";
    }
}
?>
```

Vegem com crear una instància de la classe i imprimir la constant, la variable i executar el mètode de classe.

```
$myClass = new SimpleClass();

echo $myClass->$var; // Mostrarà 300
$myClass->displayVar(); //Mostrarà 300
echo SimpleClass::MAX; //Mostrarà la constant 100.
//Noteu que no hem usat la instància de la classe.
echo SimpleClass::foo(); //Mostrarà "Hello World"
```

PHP incorpora herència simple. Per definir una relació d'herència entre classes s'utilitza la paraula clau *extends*. Vegeu el següent exemple:

```
class A {
        // more code here
}

class B extends A {
        // more code here
}
```

Per a més informació consulteu [11].

8.9 PHP + MySQL

Normalment PHP s'utilitza juntament amb un sistema de gestió de bases de dades relacional que s'encarrega de guardar tota aquella informació que canvia de forma dinàmica.
El sistema gestor de bases de dades més emprat avui en dia és el **MySQL**. El MySQL ens permet guardar grans quantitats de informació i realitzar consultes de forma ràpida i eficient mitjançant el llenguatge SQL (*Structured Query Language*).
Mitjançant el llenguatge PHP es pot accedir de forma fàcil a la base de dades per consultar, eliminar o modificar dades.
Mitjançant PHP generem codi HTML amb la informció que hi ha a la base de dades. Com que aquesta informació és dinàmica, canvía, diem que **PHP ens permet generar codi HTML dinàmic**.

8.10 htaccess

És un fitxer de configuració que utilitza el servidor Apache.
Aquest fitxer conté directives, parelles comanda/variable i el seu valor. Mitjançant aquestes directives podem configurar com ha de respondre el servidor Apache en funció de la petició HTTP que rep.
Mitjançant aquest fitxer també es pot configurar l'accés a directoris del website via autenticació HTTP.
A continuació us indiquem com configurar el control d'accés al directori principal de la web mitjançant usuari i password.

- Creeu el fitxer `.htaccess` en el vostre directori public_html.
- Aquest fitxer ha de contenir les següents directives:

  ```
  AuthType Basic
  AuthName "Entrar usuari i password"
  AuthUserFile /tdiw/tdiw-XX/public_html/.htpasswd
  Require user admin
  ```

  Cal que modifiqueu el valor de la directiva `AuthUserFile` i substituïu "XX" pel vostre grup i subgrup.

  En desar aquest fitxer en el directori `public_html` esteu restringint l'accès a aquest directori a l'usuari definit en la directiva `Require user`, per tant a l'usuari `admin`. A més cal que l'usuari `admin` especifiqui un password

que ha de coincidir amb el que està desat en el fitxer que heu indicat en la directiva `AuthUserFile`. Ara crearem el fitxer `.htpasswd`.

- Posicioneu-vos en el directori `public_html`. En aquest directori creeu un fitxer amb el password xifrat que trieu per l'usuari `admin`. Això ho farem amb la comanda:

  ```
  htpasswd -c .htpasswd admin
  ```

  Si mireu el contingut del fitxer .htpasswd que acabeu de generar veureu que el contingut té el usuari:el password xifrat. Per exemple:

  ```
  admin:$apr1$dpQ7GScT$FlHscpxxL4MXwvHjPTyCm/
  ```

- Assegureu-vos que teniu privilegis d'accés per lectura per a `others` en els fitxers `.htaccess` i `.htpasswd`.

A partir d'aquest moment quan intenteu accedir a qualsevol pàgina que estigui en el directori o subdirectoris de `public_html`, el navegador us demanarà un login: `admin` i el password que haureu configurat.
Un cop us heu identificat amb èxit, podreu navegar sense necessitar especificar de nou el login i el password durant tota la sessió de navegació.

9 Aspectes de seguretat en les tecnologies web

En la següent secció us presentem quins són els atacs més comuns als que estan exposades les aplicacions web. Us donem també consells sobre com els podeu evitar.
També us indiquem com podeu desar els passwords xifrats a la base de dades.

9.1 Atac de Cross-site Scripting (XSS)

Un atac de `XSS` consisteix en que l'atacant injecta, afegeix, codi `JavaScript` a la vostra aplicació. Aquesta injecció es pot fer de moltes maneres. Una d'aquestes maneres consisteix en posar directament codi JavaScript com a valor dels camps d'un formulari. O bé manipulant directament la URL i afegint codi JavaScript com a valor d'algun dels paràmetres de la URL.

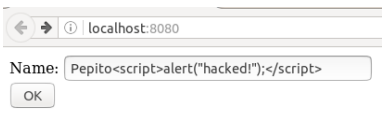
Aquí us en mostrem un exemple. Tenim el fitxer `index.html` que conté un formulari per a que un usuari especifiqui el seu nom. Quan es fa un *submit* del formulari es carrega l'escript `PHP show.php` que mostra el que l'usuari ha introduït en el camp del formulari anterior. Observeu què passa si un atacant entra en el camp del formulari el seu nom i a més a més codi JavaScript.

index.html

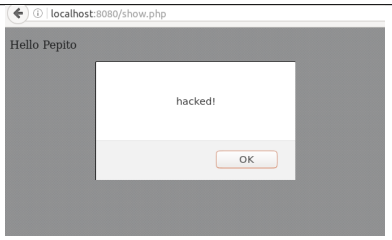
```
<!DOCTYPE html>
<html>
  <body>
    <form method="post" action="show.php">
       Name: <input type="text" name="user_name" size="30" />
      <br />
      <input type="submit" value="OK">
    </form>
  </body>
</html>
```

show.php

```
<!DOCTYPE html>
<html>
  <body>
    <p>
      Hello <?php echo $_POST["user_name"] ?>
    </p>
  <body>
</html>
```



(a) *Introducció de JavaScript maliciós en un camp de text.*



(b) *Efecte de l'execució del JavaScript maliciós.*

Com podem evitar aquest tipus d'atac?
La millor manera d'evitar aquests atacs és mitjançant la combinació de les següents funcions: `htmlspecialchars()`; `trim()` i `stripslashes()`. Visiteu [12] per veure la seva sintaxi i exemples d'ús. També en fareu ús quan implementeu la funcionalitat de validar un formulari [29].

9.2 Atac de SQL Injection (SQLI)

Aquest atac es basa en injectar instruccions SQL malicioses a través de la nostra web mitjançant, normalment, camps d'un formulari.

Amb aquestes instruccions SQL malicioses podeu arribar a llistar el contingut de taules, eliminar taules senceres, etc... Podeu trobar més informació a [30].
Per a que us feu una idea us mostrem un exemple d'atac d'`SQLI`.
Tenim un formulari de control d'accés a una aplicació web on es demana l'identificador d'un usuari i un cop es fa el submit del formulari tenim un script que mostra les dades que l'usuari ha entrat:
index.html

```
<!DOCTYPE html>
<html>
  <body>
    <form method="post" action="hello.php">
       UserID: <input type="text" name="userID" size="30" /> <br />
       Password: <input type="password" name="passwd" /> <br />
      <input type="submit" value="Login">
    </form>
  </body>
</html>
```

L'escript `hello.php` mostra les dades de l'usuari.

hello.php

```
<?php
1 $userID = $_POST["userID"];
2 $passwd = $_POST["passwd"];

3 $sql = "Select * from users where userID = '$userID' and password='$passwd'";
4 $result = mysqli_query($conn, $sql);

5 // output data of each row: The id of the user the name and the surname
6 while($row = mysqli_fetch_assoc($result)) {
7   echo "id: " . $row["userID"].
      " - Name: " . $row["firstname"].
      " " . $row["lastname"].
      " Password: ".$row["password"]."<br>";
 }
?>
```

Imagineu-vos que un atacant omple el camp del formulari amb els següents caracters:

```
UserID:  105" OR "1"="1
Password: abc" OR "1"="1
```

Quan es fa un submit del formulari i s'executa la línia 3 de l'escript `hello.php` obtenim el següent:

```
$sql = "Select * from users where userID = "105"
  OR "1"="1" AND password = "abc" OR "1"="1";
```

La condició 1=1 sempre es complirà! per tant és com si no tinguéssim cap condició. La query anterior té el mateix efecte que haver fet:

```
$sql = "Select * from users";
```

D'aquesta manera l'escript **hello.php** pintarà les dades de **tots els usuaris** de la base de dades!

Com podem evitar aquest tipus d'atac?
Hi han dues maneres. Tot i que la primera manera evitaria la major part dels atacs de SQLI encara seriem vulnerables a certs atacs de SQLI. La segona manera és infal·lible! A continuació us comentem les dues possibilitats.

- Sanejant les dades que recollim dels camps del formulari. Això s'aconsegueuix utilitzant les funcions: stripslashes() i mysql_real_escape_string(). Vegeu com funcionen a [12].
- Utilitzant consultes sql parametritzades. Vegeu com funcionen en l'explicació que us fem en la secció 9.4.

9.3 Com guardar un password xifrat a la base de dades

En el procès d'alta d'un usuari o administrador, quan s'especifica el password cal desar-lo en format hash a la base de dades. Quan l'usuari utilitza aquest password per identificar-se hem de poder comprovar que el password, introduït per l'usuari, correspon al password que ha generat el hash que tenim a la base de dades.
Aquest `hash` s'ha de calcular afegint-hi una *salt* aleatòria al password. Vegeu [9] per aprendre com fer-ho amb `PHP`.

Mitjançant la funció de `PHP` `password_hash` calcularem el `hash` d'un *pasword* utilitzant una *salt* aleatòria. El resultat d'aquesta funció el guardarem directament a la base de dades. Podeu utilitzar l'algoritme per defecte que utilitza `PHP`. Vegeu-ne la seva utilització a [8].
A l'hora de verificar un password utilitzarem la funció `password_verify`. Vegeu-ne la seva utilització a [10].

9.4 PHP + Prepared Statements

Una consulta parametritzada, *prepared statement*, consisteix en agafar una consulta SQL i fer-ne un template. Aquest template no conté els valors dels camps de la consulta. De manera que podem utilitzar el mateix template per a valors diferents.

Com s'utilizen?

1. En primer lloc construïm el template, l'esquelet de la consulta SQL. Aquí en teniu un exemple de template:

   `INSERT INTO MyGuests VALUES(?, ?, ?).`

   Fixeu-vos que no hem especificat els valors, els paràmetres, que volem insertar en la taula `MyGuests`, els hem especificat com a '?'.

2. Enviem aquesta consulta SQL al sistema gestor de la base de dades per a que la compili i l'optimitzi, sense executar la consulta.

3. Quan l'aplicació ja té els valors concrets de la consulta SQL els envia al sistema gestor de base de dades que els aplica al template que ja té compilat. El sistema gestor de base de dades executa la consulta i retorna el resultat a l'aplicació.

Quines avantatges tenen respecte una SQL tradicional?

Les avantatges més significatives de les consultes parametritzades són les següents:

- Una consulta SQL si ja està compilada serà molt més ràpida d'executar. Si aquesta consulta s'ha d'executar moltes vegades guanyarem molt de temps.

- Fixeu-vos que un cop enviat al servidor el template, només ens calen enviar els valors, els paràmetres de la consulta SQL. Això ens estalviarà bandwith si la consulta l'hem d'executar moltes vegades.

- Les consultes parametritzades són molt útils a l'hora d'evitar atacs de SQLI. Els atacs de SQLI consisteixen en afegir codi SQL com a valor del paràmetres de la consulta SQL. Si l'aplicació només ha d'enviar els paràmetres de la consulta SQL, aquest paràmetres han d'estar correctament escapats, amb el que evitem la inserció de codi SQL.

PHP ens permet programar consultes parametritzades. Nosaltres les programarem utilitzant l'objecte `mysqli` però si voleu podeu utilitzar `PDO` (PHP Data Object). Vegeu com treballar amb consultes parametritzades en [14].

Referències

[1] PuTTY: A Free Telnet/SSH Client. http://www.chiark.greenend.org.uk/~sgtatham/putty/.

[2] WinSCP, Freeware SFTP and SCP client for Windows. http://winscp.net/eng/index.php.

[3] Rebecca Murphey. Fundamentos de jQuery. http://librosweb.es/libro/fundamentos_jquery/.

[4] php.net. File Upload. http://www.w3schools.com/php/php_file_upload.asp.

[5] php.net. Function filter_var for form validation in php. http://nl1.php.net/manual/en/function.filter-var.php.

[6] php.net. Function filter_var for form validation in php. http://php.net/manual/en/filter.examples.validation.php.

[7] php.net. Function preg_match for form validation in php. http://php.net/manual/en/function.preg-match.php.

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

[9] php.net. How to store a password safely. http://php.net/manual/en/faq.passwords.php.

[10] php.net. How to verify a password's hash. http://php.net/manual/en/function.password-verify.php.

[11] PHP.net. Object Oriented in PHP. http://php.net/manual/en/language.oop5.php.

[12] PHP.net. PHP. http://php.net/.

[13] PHP.net. PHP + MySQL. http://www.w3schools.com/php/php_mysql_select.asp.

[14] PHP.net. Prepared Statements in PHP. http://www.w3schools.com/php/php_mysql_prepared_statements.asp.

[15] Javier Eguíluz Pérez. Introducción a javascript. http://www.librosweb.es/javascript/.

[16] Javier Euguíluz Pérez. Introducción a AJAX. http://www.librosweb.es/ajax/.

[17] W3C. Cascading Style Sheets. http://www.w3.org/Style/CSS/.

[18] w3c. CSS validator. http://jigsaw.w3.org/css-validator/.

[19] W3C. HTML5 HyperText Markup Language. http://www.w3.org/TR/html5/.

[20] w3c. HTML5 validator. https://validator.w3.org/.

[21] w3schools. AJAX Tutorial. http://www.w3schools.com/ajax/.

[22] w3schools. Cascading Style Sheets. http://www.w3schools.com/css/.

[23] w3schools. HTML5. http://www.w3schools.com/html5/.

[24] w3schools. JavaScript. http://www.w3schools.com/js/.

[25] w3schools. JQuery. http://www.w3schools.com/jquery/.

[26] w3schools. JSON tutorial. http://www.w3schools.com/json/default.asp.

[27] w3schools. PHP and cookies. http://www.w3schools.com/php/php_cookies.asp.

[28] w3schools. PHP and sessions. http://www.w3schools.com/php/php_sessions.asp.

[29] w3schools. PHP form validation. http://www.w3schools.com/php/php_form_validation.asp.

[30] w3schools. SQL Injection. http://www.w3schools.com/sql/sql_injection.asp.

[31] w3schools. Tag input de HTML5. http://www.w3schools.com/tags/tag_input.asp.