Índex de continguts

[Parts Pràctiques de la 1 a la 4: 2](#_Toc180586609)

[Pràctica 1: 2](#_Toc180586610)

[Pràctica 2: 3](#_Toc180586611)

[Pràctica 3: 4](#_Toc180586612)

[Pràctica 4: 5](#_Toc180586613)

[Configuració per l’us: 6](#_Toc180586614)

[(Afegit 3.0)Metodologia de treball. 7](#_Toc180586615)

[(Afegit 3.0)Actualització versió 3.0 7](#_Toc180586616)

[MYSQL 7](#_Toc180586617)

[Creació de la base de dades: 7](#_Toc180586618)

[Mysql i PHP 8](#_Toc180586619)

[Connexió PDO: 8](#_Toc180586620)

[PHP 8](#_Toc180586621)

[Creació de funcions: 8](#_Toc180586622)

[Control d’errors: 8](#_Toc180586623)

[HTML 9](#_Toc180586624)

[Construcció: 9](#_Toc180586625)

[Paginació: 9](#_Toc180586626)

[Construcció: 9](#_Toc180586627)

[Reestructuració de les pàgines pràctica 3: 10](#_Toc180586628)

Jonathan López Ramos

# Parts Pràctiques de la 1 a la 4:

## Pràctica 1:

Crear un formulari per a afegir nom, adreça de correu i text.(FET)

A l'hora d'enviar, si tot els camps són correctes, mostrarà un missatge que digui "Enviat correctament"  altrament ens informarà de quins camps estan buits. (FET)

Crear l'estructura en HTML (FET)

Crear estils CSS. Podeu fer servir els estils que us he deixat al moodle "estils.css"  o si no els que creieu convenient. (FET)

Validar les dades del formulari i enviar el correu amb PHP (FET)

tenen contingut els camps a omplir? (FET)

són les dades correctes als camps? (FET)

s'han enviat les dades? (FET)

comprovar l'estat dels errors. Hi ha errors? (FET)

evitar la injecció de codi (FET)

enviar correu configurant php.ini [0,5/4] (FET)

Feu que el vostre formulari treballi amb PHPMailer. (FET)(Tot i que va ser a posteriori ”Pràctica 4”)

1.- validació de $POST (opció més correcte) o fent required als inputs de l’html i que surti un missatge dient “Enviat correctament” (FET)(Tot i que va ser a posteriori ”Pràctica 4”)

2.- Que el contingut de les caselles no s’esborri en cas d’haver algun error(FET)

3.- diferenciar la lògica de la vista(FET)

## Pràctica 2:

Crear una BDD MySQL des de phpMyAdmin(FET)

Crear una taula per a enregistrar articles on emmagatzemarem ID, titol, cos. (FET)

Una vegada creada la taula d’articles s’ha de poder gestionar, és a dir, s’ha de poder **inserir, modificar, esborrar i/o consultar els articles** d’aquesta taula. Realitzar-ho mitjançant *Prepared Statements!!* (FET realitzar amb codi javascript per tal de no tenir molts mes formularis de vista)

Tots els fitxers php i html han de contenir el nom i cognom de l’alumne. (FET)

El codi ha d’estar comentat. (FET)

Ha de funcionar correctament les opcions Inserir, Modificar i Esborrar. (FET)

Les cerques s’han de fer al menys per DNI. (FET però es realitzen per ID únic)

Els fitxers que genereu s’han de separar la lògica de la vista, és a dir, creeu una estructura com per exemple aquesta.(FET)

## Pràctica 3:

La informació que mostrarem a cada pagina serà extreta de la BDD i aquesta serà el Títol de l'article i el seu cos. (FET)

Connecteu-vos a la BDD mitjançant *try...catch* i fent servir *PDO*. Comproveu si la connexió és correcte i feu servir *die()* en cas de ser necessari. (FET)

L’extracció de les dades la fareu mitjançant una consulta SQL i en cas de no haver-hi articles, redirigireu a la vostra url de la pràctica que es troba a localhost. (FET)

Establir el número de pàgina en la que l’usuari es troba, en cas de no trobar cap valor serà per defecte la pàgina 1. (FET)

Definiu 5 post per pàgina i calculeu el número total de pàgines que tindrà la paginació en funció del total dels articles. (FET)

Creeu dos etiquetes *div*, en una d’elles guardarem els articles i en l’altra tractarem l’estat del botó *Anterior* i *Següent.* (FET)

En cas de crear una BDD, has de lliurar l’esquema de la mateixa amb el següent format: ***pt03\_Nom\_Cognom.sql***(FET)

A cadascú dels fitxers desenvolupats haurà d’aparèixer el vostre nom a la capçalera i el codi comentat. (FET)

Cal fer un README tot especificant i/o detallant què heu fet a la pràctica. Justifiqueu els motius(FET)

Adjunteu l’esquema de la vostra BDD. (FET)

El nom del fitxer haurà de ser: **Nom\_Cognom\_Pt03.rar/.zip**

## Pràctica 4:

En obrir el web sortiran tots els articles de forma pública sense excepció i sense opció d’inserir, editar o eliminar articles. (FET)

A la mateixa pàgina principal trobarem una opció per a logar-se o enregistrar-se. (FET)

Crear un formulari de inici de sessió i registre d’usuaris en una base de dades i posteriorment verificar quan inicien sessió els usuaris de la base de dades i llavors permetre’ls editar el contingut (els post de la paginació) si no iniciem sessió (usuari anònim) permetrem només veure els articles. Mireu com ho faríeu tenint en compte que el tractament de les dades que hi ha a la paginació no serà el mateix si ens validem que si no estem validats. (FET pots veure tots els articles sense estar loguejat, un cop loguejat et porta a un altre formulari on pots rectificar dades de la base de dades.)

S’haurà de comprovar si l’usuari ja existeix en qualsevol dels dos casos (inici sessió o registrar-se) i validar la contrasenya demanant-la dues vegades (registrar-se) (FET “l’alteració del que es veu s’ha realitzat amb codi Javascript login/registre” en inici de sessió es realitza únicament usuari(que es el correu) i en registre, nom, correu, contrasenya(s’ha d’introduir dues vegades).

S'ha de tenir en compte la fortalesa de la contrasenya, a més aquesta s'haurà de guardar encriptada. (FET mínim 8 caràcters.)

Si l’usuari comet un error a l’hora d’enregistrar-se, s’hauran de mantenir les dades al formulari actual. (NO FET encara falta)

Una vegada validat, s’ha de mantenir la sessió de l’usuari almenys 40min activa. (NO FET encara falta) (FET tot i que no es la forma mes segura, ja que t’acaba expulsant de la sessió si passa molta estona, o realitza sessionstart sol... però d’aquella manera.....)

Cal treballar amb sessions i cookies. Valoreu en quins casos cal treballar amb sessions i quan amb cookies(FET sessions)

Creeu una opció per a recuperar la contrasenya des del mateix formulari. (FET es realitza amb un token que dura 1 hora.)

Cal treballar amb repositori Git(FET per fer-ho mes còmode es realitzarà dins de carpetes de Versions.)

A cadascú dels fitxers desenvolupats haurà d’aparèixer el vostre nom a la capçalera i el codi comentat. (FET tot i que encara falta codi per comentar.)

Cal fer un README tot especificant i/o detallant què heu fet a la pràctica. Justifiqueu els motius(FET mes o menys tot i que no se si es el que es demana.)

Adjunteu l’esquema de la vostra BDD. (FET)

# Pràctica 5:

**# Recuperació/canvi de contrasenya**

Donarem a l'usuari dos opcions de canvi de contrasenya:

Si està logat, permetre'm que pugui accedir des de la mateixa web i fer un canvi de contrasenya introduint la contrasenya antiga i dos vegades la seva contrasenya. (FET)

Imatge que conté text, captura de pantalla, Font, nombre

Descripció generada automàticament

Si no està logat i no recorda la contrasenya i per tant vol recuperar-la, llavors donarem l'opció a l'usuari que demani la recuperació de la contrasenya introduint el e-mail amb el que està donat d'alta a la BDD. Aquest correu electrònic proporcionarà un token amb una validesa màxima. (FET)

Imatge que conté text, captura de pantalla, Font

Descripció generada automàticament

Si decidiu fer recuperació de contrasenya d'alguna altra manera, cal que ho justifiqueu al README.(FET)

**# Ordenació dels articles**

Ordenar articles de forma DESC o ASC, en funció de valors com ara la data o alfabèticament.(FET)

Imatge que conté text, Font, captura de pantalla

Descripció generada automàticament

**# Remember me**(FET)

Si l'usuari marca aquesta opció en logar-se, haurem de recordar la seva contrasenya quan iniciï sessió. Cal fer servir un token per a tractar aquest cas. (FET)Imatge que conté text, captura de pantalla, Font, línia

Descripció generada automàticament

**# reCAPTCHA**

**Inicia Sessió Registra't**(FET)

**Correu electrònic**(FET)

**Contrasenya**(FET)

Quan l'usuari prova de logar-se, i després de 3 intents fallits se li demanarà a l'usuari que confirmi un reCAPTCHA. El tipus de reCAPTCHA que utilitzeu serà de la vostra elecció. .(FET)

Imatge que conté text, captura de pantalla, Font, nombre

Descripció generada automàticament

**Clau html: 6LdM4nQqAAAAAHi\_d\_9xUdLLX1lx\_aLwJP8UlTYS**

**Clau servidor: 6LdM4nQqAAAAACaBSmZS2fyjtlKs21Sa0RXPX8qq**

(FET)

**# Autenticació social**

ToDo...................OAuth i HybridAuth

**# Editar perfil personal**(FET)

Canviar dades del perfil personal, com ara la imatge/avatar, el nickname, .... (FET)

Les dades actuals que pugui modificar l'usuari, li han d'aparèixer al formulari de canvi de dades del perfil. (FET)

Imatge que conté text, captura de pantalla, Font, línia

Descripció generada automàticament

**# Usuari Admin**(FET)

Crear un usr Admin que pugui esborrar altres usuaris. Què cal fer amb els articles associats a aquest usuari esborrat? Justifica la teva elecció al README.

**# Barra de cerca**(FET)

Permet cercar per almenys un dels camps de les fitxes. Guarda les cerques realitzades.

La forma senzilla de fer-ho és, que una vegada introduïm el valor a cercar, premem enter i el cerca, o podem fer que a mida que anem cercant va mostrant els resultats de cerca (caldria fer javascript/ajax) .(FET)

**# Configuracions de seguretat**

Deixa constància al README de les configuracions de seguretat que apliqueu, entre d'altres al fitxer .htaccess >> diapo 81

# Configuració per l’ús:

**Botó config:**

Imatge que conté text, captura de pantalla, Font, nombre

Descripció generada automàticament

**Httpd.conf:**



**Ctrl + B dins l’arxiu:**

Imatge que conté text, captura de pantalla, pantalla, nombre

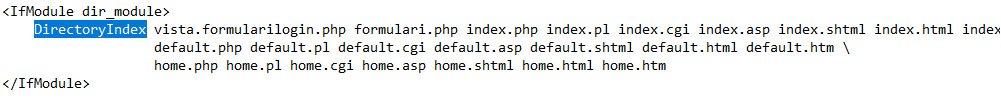
Descripció generada automàticament

**Escriurem DirectoryIndex:**

Imatge que conté text, captura de pantalla, Font, pantalla

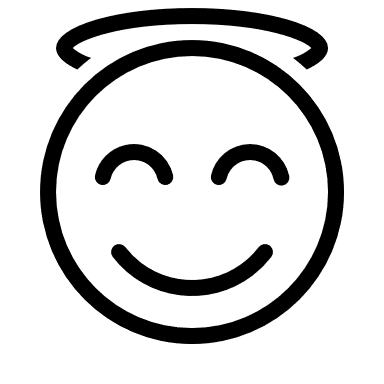
Descripció generada automàticament

**Dins escriurem el nom corresponent al nostre directory index, en el nostre cas:**



**Vista.formulari.login**

**Un cop fet això realitzarem la tasca de executar la base de dades.**

**Ja hauria de funcionar!!**

# Metodologia de treball.

Quan s’està aprenent i a més no es comte amb memòria suficient... com es el meu cas, s’ha d’establir un mètode que tot i que es molta mes feina, al final crec que per a mi es millor.

Normalment intento realitzar la vista conjuntament amb les funcions i de mes necessari per tal de que la pàgina funcioni correctament.

Un cop arribat al correcte funcionament, aprofito per tallar codi i generar el controlador corresponent, d’aquesta manera, verifico que el codi esta bé o relativament ve i el que falla es la ruta(ja que de vegades necessita una ruta o un altre, aquest tipus de codi no m’acaba de funcionar correctament) o les crides corresponents.

# Actualització versió 3.0

L’usuari ja pot veure les compres a realitzar, s’han separat els codis de l’html de tal manera que ja es pot veure clarament que hi ha a controlador que hi ha a la interfície Web.

Tot i que encara no es vistos per l’usuari, s’ha creat un sistema d’errors correctes i errors, d’aquesta manera recollim al sessions corresponent i printem a cada pàgina l’error o encert corresponent.

# Actualització 4.0

L’estructura d'aquests sistemes de fitxers a data d'avui es:

* **htdocs** inclou(vista)**:**
  + **vista.formulari.php**(Encarregat de visualitzar els articles, modificar-los i comprar)
  + **vista.formularilogin.php**(aquest es el primer per defecte que visualitza la web)
  + **vista.oblit.php**(encarregat de veure a l'hora de recuperar la contrasenya)
  + **vista.reestableix\_contrasenya.php**(Encarregat de la vista a l'hora de recuperar la contrasenya amb el token del correu)
  + **vista.veureCompres.php**(que conté la visualització de les compres realitzades per l'usuari)
* **també a dins de htdocs:**
* La carpeta **controlador** dins de **htdocs** conté:
  + **cont.compres\_usuari.php(**conte el necessari per tal de modificar la base de dades i afegir les compres realitzades per l'usuari)
  + **cont.connexio.php**(conté la connexió a la base de dades i retorna **$pdo** amb la funció **obtenirConnexio**)
  + **cont.logout.php**(conté una funció que es crida a l'hora de clica el botó de logout i redirigeix a **vista.oblit.php**)
  + **cont.processar.php**(conte funcions i de mes, que s'utilitzen per a mostrar, fer l'update, delete i modificar de **vista.formulari.php**).
  + **cont.tokenRecCon.php**(encarregat de alterar la base de dades i comparar el token en cas d'estar caducat a mes, modifica la contrasenya de l'usuari.)
  + **cont.veureCompres.php**(conte les dades necessàries per tal de mostrar les dades a **vista.veureCompres.php**)

**també a dins de htdocs:**

* tenim la carpeta **model** aquesta conté:
  + **model.login.php**(encarregat de agafar las dades de login i redirigir a vista.formulari.php)
  + **model.oblit.php**(encarregada de rebre les dades d'email de l'usuari que no recorda la contrasenya i utilitzar la carpeta phpmailer que tenim a dins de htdocs, redirigeix al enviar el formulari de les dades a vista.formularilogin.php)
  + **model.register.php**(encarregat de fer l'ingrés dels usuaris a la base de dades).

Imatge que conté text, captura de pantalla, Font

Descripció generada automàticament

**S’ha afegit una política de coockies, de tal forma que si no acceptes no es pot veure la pàgina de vista.formularilogin.php ni registre amb una combinació de PHP i Javascript.**

**S’han reparat un munt d’errors per culpa de les rutes, que a mi em funcionaven però en un altre ordinadors no, com imatges, require, etc.**

**La recuperació de contrasenya es realitza des de un altre taula de la base de dades, així, si tenim un alt numero d’usuaris no sobrecarreguem la taula de login i registre al recuperar la contrasenya.**

**Com la recuperació es realitza mitjançant un token amb caducitat d’una hora, ens assegurem que dit toquen l’utilitza la persona del correu corresponent.**

**A l’hora de printar l’expulsió per inactivitat no s’ha aconseguit que aquest error sorti, encara no se per què....(S’ha de solucionar, clar al destruir la sessió no s’envia res... mirar a veure com solucionar això....).**

**S’ha establert que l’usuari no pot estar mes de 40 minuts a la sessió, aquesta funció s’ha posat a l’arxiu de connexió.php, per tal de que per cada vegada que es realitzi una acció, aquesta, reiniciï el comptador de la funció que controla la sessió.**

# (Actualització)Versió 5.0

## Tasques no implementades de treballs anteriors:

### Practica 1

Que el contingut de les caselles no s’esborri en cas d’haver algun error. (FET)

### Pràctica 2

El codi ha d’estar comentat. (FET)

### Pràctica 3

Tot (FET).

### Pràctica 4

Si l’usuari comet un error a l’hora d’enregistrar-se, s’hauran de mantenir les dades al formulari actual. (FET)

Una vegada validat, s’ha de mantenir la sessió de l’usuari almenys 40min activa. (FET, cada acció que requereixi d’una connexió a la base de dades, reinicialitza el comprador.)

Cal treballar amb sessions i cookies. Valoreu en quins casos cal treballar amb sessions i quan amb cookies(FET S’ha realitzar el típic anunci de política de coockies a l’entrada del formularilogin, si l’usuari no accepta les coockies no pot visualitzar el contingut)

# (Afegit)Pràctica 5

Imatge que conté text, captura de pantalla, programari

Descripció generada automàticament

* Afegida una nova vista, en aquest cas **vista.canviContUser.php** amb la que l’usuari pot accedir des de l’apartat compres un cop logat.
  + Aquesta vista demanarà l’antiga contrasenya i demanarà dos cops per a la que es vol canviar.
  + Per tal d’augmentar la seguretat de les contrasenyes s’ha establert que aquestes com a mínim tinguin 8 caràcters i a més, han de contenir números, lletres i caràcters especials.

S’ha comprovat que la forma de logar amb la funció que es troba a **vista.connexio.php** funciona de la forma correcta, ja que reinicialitza el comptador cada cop que s’accedeix a la base de dades per alguna cosa.

S’està establint de la mateixa manera a tots els documents de la vista, una forma de que si l’usuari no té la sessió iniciada, reenvií al formulari login. D’aquesta forma s’aconsegueix tenir una mica de seguretat de que l’usuari no pot establir la connexió a cap document si no està logat.

S’ha canviat la localització dels errors i encerts de les funcions, per tal de mostrar-les una mica millor, ara es poden visualitzar a una finestra que es troba a la dreta(logout-container).

* S’ha afegit al codi de **model.login.php** el **recaptcha v2** corresponent, aquest, mira si els intents son superiors a 3, per tal de que un cop entres tres cops la contrasenya malament, al quart intent es mostrarà el recaptcha corresponent. Aquesta acció s’ha realitzat amb combinació de codi, php i javascript.
  + S’ha creat una nova carpeta on establir les keys de server i client en una variable de sessió per tal de donar una seguretat a les keys correstponents.

S’ha canviat el document **model.oblitcontra.php**, per tal de que les select es facin amb la id de l’usuari en comptes del per el correu, llavors el que fem es realitzar les selects corresponents per tal de veure la id de l’usuari al que correspon el correu.

S’ha rectificat el codi corresponent de la base de dades per tal d’utilitzar un altre taula per el token del remember me.

S’han canviat a més les dades de la mateixa per tal de que una taula s’encarregui de la gestió de tokens per a l’inici de la sessió recordant la contrasenya.

Per tal de dividir codi ara **cont.processar.php**, s’ha traslladat part del codi i funcions a **cont.articles.php**, aquest, s’encarregarà ara de mostrar els articles d’una forma o un altre, a més, cridarà a **cont.processar.php** que conté la connexió a la base de dades i **l’startsession,** a **vista.formulari.php** es farà la crida a **cont.articles.php**.

Ja es possible mostrar articles de forma ascendent i descendent, a més de buscar i mostrar un sol article. **Vista.formulari.php** crida a **cont.articles.php**, i aquest retorna les variables carregades al nom de sessió corresponent a **vista.formulari.php**, aquest crida a les funcions corresponents per tal de mostrar el demanat, ja sigui cerca per nom, o mostrar per ordre ascendent o descendent(**nom**).

S’afegeix també una forma d’esborrar l’article comprat corresponent a **vista.veureCompres.php**, crida amb un formulari a **cont.elimarCompra.php**, aquest s’encarrega d’accedir a la base de dades per tal de que esborri l’article corresponent a traves de l’id corresponent, ja que d’un altre forma esborraria possiblement compres d’un altre usuari(per exemple).

Per tal de fer més fàcil l’ús de la pàgina, s’ha creat una redirecció amb un **index.php** per tal de redirigir automàticament a **vista.formulari.php**, d’aquesta manera no s’ha de configurar res.

S’ha pujat la nova versió a donDominio:

<https://www.jlopez5.cat/>

[jhonlopezramos@gmail.com](mailto:jhonlopezramos@gmail.com) “Administrador” contrasenya Abcd1234

[gabrielgusman@gmail.com](mailto:gabrielgusman@gmail.com) “Ususari pla” contrasenya Abcd1234

a data de 14/11/2024 s’ha rectificat el codi per tal de que el MVC estigui on toca. S’han rectificat noms per tal de que estiguin tots amb la mateixa construcció, nova paraula dins del nom, amb majúscules.

S’ha afegit diversos directoris git per tal de fer la pujada dels arxius mitjançant un token.

Ampliació de vites i constants, s’ha afegit una **vista.administrador.php,** aquesta nomes es accessible per un usuari de nivell 1, s’utilitza una variable de sessió per aquesta tasca. Aquesta vista utilitza una llista que es crida al entrar a un document anomenat **cont.administracioUsuaris.php**, a on l’administrador te privilegis per esborrar(canviar el nivell dels usuaris “administrador o no”, encara no està implementat del tot).

La taula usuaris a sigut modificada també per a realitzar aquesta tasca, s’ha creat una columna nova anomenada **nivell\_administrador** en la que es un int, el valor per defecte serà 2, per tant el nivell 1 serà administrador i l’altre usuari pla.

S’ha creat una nova taula per les imatges corresponents als usuaris, d’aquesta manera no tenim redundància de dades a la taula d’usuaris, aquesta nova taula emmagatzema la ruta de les imatges que poden escollir de perfil.

Per aquesta tasca s’han creat la **vista.edicioPerfil.php**, que conté la vista per tal de canviar a la base de dades les dades corresponents, tant la ruta com el nom d’usuari. Les dades s’envien al model(**model.edicioPerfil.php**) i aquest utilitza les funcions que tenim al controlador(**cont.edicioPerfil.php**).

S’han afegit les dades corresponents a cada una de les vistes per tal de que es puguin veure les imatges de l’usuari.

Ojo cuatro conceptos chorra:

Autenticació social:  
va con token que proporciona github, twiter etc.

Guarda les dades a la nostra base de dades.

Una nova taula que es token+des de on arriba l’usauri

Uno de cada un outh2(OpenAuthorizathion)

Llibreria de google google-api-php-client

S’ha de posar que si esta loguejat per autorització social no pugui canviar el password, aquest camp s’ah d’ocultar

S’ha de mirar si esta a un lloc o l’altre i preintar si es un login normal o es autenticació socila i en quina xarxa social es trova

Al git hub developersetings s’ha de canviar la url corresponent per tal de que la autenticació social funcioni

# MYSQL

## Creació de la base de dades:

Primer de tot vaig comença mirant que podia necessitar per tal de crear una **base de dades a Mysql.**

Com es una base de dades d’articles, poden ser per exemple una tenda de fruites, llavors vaig crear la base de dades únicament amb un **id, nom i cos**.

El id\_articles es un autoincremental de clau primària.

El nom correspon al nom de la fruita que s’ha de posar.

El cos es una descripció de la fruita, que podria ser d’on prové per exemple amb la seva denominació d’origen.

Vaig arribar a la conclusió que la pràctica era petita, llavors, vaig fer una taula de tres columnes, **Cos(que inclou text sobre l'aliment), nom(que es el nom de la fuita) i evidentment la id que correspon auto-incremental.**

# Mysql i PHP

## Connexió PDO:

És van realitzar proves pertinents per a establir la connexió a la base de **dades amb PDO**.

# PHP

## Creació de funcions:

Es va crear la funció **obtenirArticles** que es la responsable d’emplenar la llista que surt sota del formulari.

També vaig fer la funció **inserirArticles** que depèn de l’anterior funció(en part), si retorna un numero superior a 0 no realitza l’entrada de dades corresponent i avisa de l’error mitjançant la URL. Aquesta mateixa funció també mira si els camps es troben o no plens.

Era necessari també crear una funció **modificarArticle** per tal de com be diu el nom de la funció, modificar alguna dada.

La funció **esborrarArticle** s’encarrega d’esborrar l’article mitjançant la id. Si no existeix id, retorna l’error.

## Control d’errors:

Totes les funcions miren si els camps tenen o no dades, si no s’introdueix res envia el missatge d’error corresponent.

Es va crear també una funció per tal de veure si existeix o no a la base de dades el nom corresponent anomenada **existeixArticle** per tal de no tenir articles repetits.

També es va crear una funció **existeixArticlePerId** que com el seu nom indica es per mirar si la ID es troba a la base de dades, l’estructura es la mateixa que a la funció **existeixArticle**, però la seva sentencia sql es diferent.

A les dues, si es troba un numero superior a 0 retorna **true**.

A l’hora de pujar les dades vaig veure que solament deixava pujar noms o text sense cap tipus de **simbologia de l’escriptura en català**(accents, dièresi, apòstrof...). Llavors es va alterar la expressió regular per tal de deixar passar aquestes dades.

Finalment es va afegir el **try catch** corresponent a cada una de les funcions. Cosa que temps després vaig veure que no s’utilitzava de la forma que creia.

Per tal de mostrar errors o encerts a l’hora de realitzar les tasques, es va posar també a les funcions, que retornés **l’error o “encert”,** que en aquest cas **la paraula correcte es aconseguit.**

Al final del programa al rebre el **POST**, tenim molta informació, ja que arriba si es **Update, delete o insert** i depengen d’aquesta arribada, podem agafar els errors de la funció corresponent i afegir-ho a una variable, en el nostre cas utilitzem dos, una que es **missatgeError** i l’altre que es **missatgeCorrecte.**

Al final de tot es va afegir **$\_SESSION** per afegir els missatges d’error i poder portar-los cap al formulari.php.

De tant en tant intentava anar canviant una mica la vista(**frontend**), per tal de veure una cosa una mica millor.

# HTML

## Construcció:

Com ja teníem la taula de la base de dades, sabíem que necessitàvem un formulari que tingues com a mínim tres camps input, amb un boto d’enviament.

Un cop creada l’estructura havíem de mostrar de forma dinàmica el contingut del formulari per tal de no tenir varies pàgines.

Vaig trobar que existia una forma mitjançant **Javascript** d’utilitzar un únic document per tal de mostrar diferents dades**(formulari dinàmic).**

Es va crear una funció anomenada **mostrarFormulari**, aquesta, el que fa es activar o desactivar la **capça div** de l’**HTML** on es troba **el label i l’input** mitjançant la **id corresponent** i alterant el camp **Style** on podem mostrar o no, depengen del **Radio o input** clicat.

**Inserir(Insert):** Mostra l’entrada de dos inputs, Nom i Descripció, per que l’altre recordem que es auto-incremental.

**Modificar(Update):**Per a poder modificar s’ha de posar la ID, el nom i la descripció corresponent.

**Esborrar(Delete):** Elimina l’article amb la ID.

Es va afegir un petit text explicatiu del que es pot fer a cada botó del formulari amb el **placeholder**, que s’utilitza per posar text als inputs, o mitjançant **title** al botó que explica que puja les dades.

Es va afegir una taula que cada cop que es recarrega la pagina realitza la consulta i empena la taula.

# Paginació:

## Construcció:

En principi no sabia ni el que es referia el terme paginació... tot i que es van mostrar exemples a classe no sabia ni per on començar.

Un cop mirat per internet com realitzar-ho, vaig alterar la funció obtenir articles per tal de que agafés la variable que tinc a formulari php del numero de items a mostrar per pàgines(aquest es pot posar tranquil·lament amb un input més endavant per tal de que l’usuari esculli quants vol veure.).

S’ha creat també la funció obtenir total d’articles i obtenir total de pagines per tal de fer la crida al bucle que s’encarregarà de repetir fins que arribi a 5.

S’ha fet el corresponent try catch a cada una de les noves funcions.

S’ha afegit també que es pugui veure la pagina on et trobes al header(URL) de la pagina. I també el session\_destroy a final del document processar php.

S’han afegit les crides corresponents a les funcions amb els comentaris de que fa cada una de les crides.

S’ha rectificat la base de dades per a que emmagatzemi la ruta de les imatges i aquestes s’han posat al lloc corresponent, la carpeta img.

S’han afegit de moment un numero total de 20 fruites i verdures per veure que la paginació funciona correctament.

A més, s’ha des habilitat el botó d’anar cap enrere si et trobes a la pàgina inicial.

# Reestructuració de les pàgines pràctica 3:

Arriba un moment que hi ha tant de codi que ja no se ni on va que i més quan es passa d’un llenguatge de programació a un altre, llavors, el que faré serà canviar les pàgines i a partir d’aquí crear un arxiu per a cada una de les pàgines.

D’aquesta manera es molt més fàcil la vista ràpida e intuïtiva sobre les dades.

La reestructuració serà que el login i registre seran la pagina principal, si fas login o registre et mostrarà el formulari.php amb les condicions que teníem avanç.

La intenció es no saturar de codi cada arxiu, ja que quan fa temps que no ho mires trigues molt a mirar que pot estar passant.

Imatge que conté text, captura de pantalla, Font

Descripció generada automàticament

De tant en tant surt un error a l’arrancada de Mysql que es pot solucionar amb:

Solució [aquí](https://www.youtube.com/watch?v=5cCT1ThLqqU).

Contrasenya d’aplicació: **yndf uqmg dhxv crii**

Model Vista controlador

**Vista tota la part que es mostra a l’usuari**

Nomdelarxiu.**view**.php

**Controlador solament es fan include.**

En el processar.php (Controlador)s’ha d’incloure Sessions i cookies

Include

Include\_once

Require

Require\_once

Tasques a realitzar de moment per a l’entrega de cookies i sessions:

Existeixen un munt d’errors a solucionar:

Treballar amb cookies.

Canviar els noms dels arxius o fitxers per a saber mes ràpidament si es tracta de view o gestió de codis.

Model Vista Controlador...

**Controlador solament es fan include.**

Imatge que conté text, pissarra blanca, escriptura a mà, a cobert

Descripció generada automàticament

Mostrar a la pàgina de login els diferents articles dins de la pagina corresponenst. FET

Separar codi de la pagina formularilogin per tal de deixar el mínim possible.