Grado en Ingeniería Informática
Universidad de Granada

Práctica 2: Programación Web

Víctor Bricio Blázquez
vbricio@correo.ugr.es

11 de junio de 2020

1. Introducción

Para comenzar, me gustaría indicar que siempre que he tenido que hacer un cambio de página en la página web, hay dos posibilidades, y he hecho el mismo proceso, es el siguiente:

- Caso en el que no hay que hacer ninguna interacción con la base de datos:
  - Gracias a la variable de tipo array $_SESSION voy guardando variables como el nombre del usuario o el nombre, el color y la imagen de la biblioteca actual en la que está el usuario.
  - Las variables que no se puedan guardar en $_SESSION, como por ejemplo cuando estamos en el archivo gestion.php y queremos acceder a bd1.php, tendremos que elegir una de las bibliotecas. Esa variable no se puede considerar *global* en el cambio de archivo.
    Cuando pasa esto lo que hago es hacer un **form**, con el action igual al archivo al que se dirige, y con el método post, que tiene inputs de tipo *hidden* para poder conservar estas variables (como el nombre, el color y la imagen de la biblioteca actual, cuando no se conocían en el archivo anterior).
- Caso en el que hay que hacer alguna interacción con la base de datos:
  - En este caso primero hago un **form**, con action un nuevo archivo, por ejemplo, si se está tratando el alta de una sección (desde altaseccion.php) el nuevo archivo se llamará altaseccongestion.php, y con el método post, con todas las variables a introducir en en la base de datos y con las variables que mencioné antes como inputs con tipo *hidden*.
  - Una vez que estamos en este nuevo archivo, que es completamente PHP, trataremos la gestión que corresponda. En el ejemplo que he puesto antes se trataría la gestión del alta de la sección.
    Creo un vector con los datos que se deben introducir en la consulta y realizo la consulta. Si es para introducir un nuevo elemento haré un **INSERT**, si hay que eliminar un elemento haré un **DELETE**, y si debo modificar un elemento haré un **UPDATE**.
  - Tras esto, conservo las variables necesarias en $_SESSION y hago un *echo* con el siguiente contenido:
    Un documento HTML vacío que lo único que tiene es una función en javascript que lo que hace es ir al archivo adecuado (según lo que estemos haciendo) y se ejecuta en la carga del body, utilizando la función **onload()** del body.
  - De esta forma se puede se puede modularizar mucho mejor, a a la hora de detectar errores es más sencillo encontrar el origen.

2. Base de datos

Las tablas que tengo en mi base de datos son la siguientes:

- Tabla **Usuario**: Los atributos de esta tabla son:
  - **usuario**: es el nombre del usuario.
  - **contrasenya**: es la contraseña del usuario.
  - **telefono**: es el teléfono del usuario (inicialmente lo pongo a 0 para que sea un número, el usuario podrá modificarlo tras registrarse).
  - **correo**: es el correo del usuario (inicialmente estará vacío, el usuario podrá modificarlo tras registrarse).
  - **web**: es la página web del usuario (inicialmente estará vacía, el usuario podrá modificarlo tras registrarse).
- Tabla **Biblioteca**: Los atributos de esta tabla son:
  - **titulo**: es el nombre de la biblioteca.
  - **color**: es el color que se va a utilizar en la biblioteca.
  - **imagen**: es la imagen que va a representar a la biblioteca, esta es la imagen que se verá siempre que se esté en una subpágina de la biblioteca.
- Tabla **Seccion**: Los atributos de esta tabla son:
  - **titulo**: es el nombre de la sección.
  - **biblioteca**: es la biblioteca a la que pertenece la sección.
  - **descripción**: es la descripción de la sección.
- Tabla **Recurso**: Los atributos de esta tabla son:
  - **titulo**: es el nombre del recurso.
  - **autor**: es el autor del recurso.
  - **seccion**: es la sección a la que pertenece el recurso.
  - **tipo**: es el tipo de dato que es el recurso (texto, imagen, vídeo o audio).
  - **metadato1**: es un dato extra que se quiera guardar del recurso.
  - **metadato2**: es otro dato extra que se quiera guardar del recurso.
  - **desctipcion**: es la descripción del recurso.
  - **imagen**: es la imagen del recurso.
  - **biblioteca**: es la biblioteca a la que pertenece el recurso.
  - **id_seccion**: este será el id que tenga el recurso dentro de la sección. Esto se usará en el archivo recursoseccion.php, para saber qué recursos están en la sección actual. Y luego, cuando vayamos al archivo recurso1.php, que podamos hacer siguiente y anterior de forma adecuada.
  - **id_biblioteca**: este será el id que tenga el recurso dentro de la biblioteca. Esto se usará en el archivo bd1.php, pues en la parte de la derecha pondremos los últimos tres recursos que se hayan añadido. Con esta columna podremos saber siempre cuáles han sido los tres últimos recursos.

3. Elementos repetidos en los archivos

- Hay varios archivos en los que hay que mostrar todas las secciones de la biblioteca. Por supuesto, depende de la biblioteca en la que estemos habrá unas secciones u otras. Para controlarlo siempre que se debe hacer esto ejecuto una consulta a la base de datos y tomo todas las secciones que tienen como atributo **biblioteca** la biblioteca actual (que es una de las variables voy guardando a través de los archivos).

- Tanto en el archivo index.php como en gestorbd.php hay una función hecha en javascript que hace los siguiente: cuando se pasa el ratón por encima de la imagen de alguna de las bibliotecas veremos un alert que nos dice de qué biblioteca se trata y qué recursos tiene.

- Principalmente en el archivo datospersonales.php, pero en otros como la edición y alta tanto de bibliotecas, secciones y recursos, he utilizado funciones javascript que controlan los tipos de datos que se introducen en los formularios y el tamaño de los mismos.
  Por ejemplo, se puede añadir un número de teléfono, este debe tener exactamente 9 dígitos, si no son 9 o alguno de los dígitos es una letra o un caracter extraño se verá un alert que nos informará del error correspondiente. Lo mismo pasará si introducimos una nueva contraseña con menos de 3 caracteres o mas de 9, por poner otro ejemplo.

- Siempre que queramos borrar un recurso o una sección, aparecerá un desplegable con las opciones que tenemos para eliminar. Si estamos en una biblioteca no podremos eliminar ni recursos ni secciones de una biblioteca distinta. No se nos ofertará esta posibilidad en el desplegable.
  Lo mismo ocurrirá cuando queramos dar de alta o editar un recurso, a la hora de darle una sección. Nos aparecerá una desplegable con las distintas secciones posibles (para añadirlo a ésta) que haya en la biblioteca actual.

- En los archivos en los que se crea o se edita tanto una biblioteca como un recurso (en la parte de la gestión de la operación), debería haber una parte en la que se sube una imagen al servidor para que el usuario pueda elegir la imagen para representar al recurso o biblioteca.
  Sin embargo, después de hacer varias pruebas he seguido obteniendo un error de permisos que no he sabido arreglar.
  Los pasos que sigo son los siguientes:
  - Creo una variable (*$target_dir*) que hará referencia al path en el que se almacenaría la imagen. En mi caso es una carpeta llamada **imagenes** en la carpeta inmediatamente superior al archivo.
  - Obtengo la variable *$uploadPath* que será el path completo de la imagen a subir. Se calculará como la unión entre la variable anterior y el nombre de la imagen.

El nombre lo podremos encontrar en *$_FILES[ïmagen"]["name"]*, siendo *imagen* el atributo *name* del input con *type=file* que teníamos en el formulario.
A esta variable también he probado añadirle al principio el path actual, pero ninguna de las dos ha sido efectiva.

- Tras la creación de estas variables se debería usar la función *move_uploaded_file* cuyo parámetros son el path temporal del archivo que queremos subir al servidor, y el path al que queremos subirlo.
  El path temporal en el que se encuentra nuestra imagen está en *$_FILES['imagen']['tmp_name']* y el path donde la queremos subir es la variable $uploadPath.
- He buscado por Internet y en todos los sitios web han coincidido que esta era la forma de proceder, también he intentado subir la imagen al servidor betatun, pero tampoco he obtenido respuesta satisfactoria.

- Este procedimiento está comentado al principio del archivo *crearbdgestion.php*, la intención era que funcionara aquí, para luego poder replicar el código en el resto de archivos que requirieran esta funcionalidad.

Por último, solo me queda añadir que me ha parecido mucho mas elegante utilizar HTML junto con PHP y MySQL, que utilizar únicamente HTML. Con lo que he aprendido en esta práctica considero que todo se puede modularizar y generalizar mucho más.