

UD6-UD7. Pt1 – Orientación a objetos y acceso a BD con PHP (1)

En el aula virtual tenéis el proyecto Pt1_version_0.

- ⑩ Está hecho con arquitectura MVC, empleando la librería **MySQLi** para acceder a BDD
- ⑩ Es una extensión de PHP que permite acceder a Bases de datos MySQL.
- ⑩ Tutorial introductorio:

https://www.w3schools.com/php/php_mysql_connect.asp

- ⑩ Tutorial oficial MySQLi

<http://php.net/manual/en/book.mysql.php>

- ⑩ El patrón **Modelo-Vista-Controlador** estructura la aplicación en 3 capas.
- ⑩ Según este patrón, los controladores reciben las peticiones de los formularios que hay en la vista, elaboran una respuesta a partir de los datos del modelo, y muestran el resultado al usuario a través de una vista.
- ⑩ Habitualmente, la capa Modelo se divide en dos subcapas, Business y Persistence.
- ⑩ En Business encontramos las clases con las que trabajan directamente los controladores (DTO's, Data Transfer Objects).
- ⑩ En Persistence se encuentran las clases que permiten persistir los cambios en la BDD. Son los DAO's (Data Access Objects). Lo que se hace es pasar a los objetos DAO los objetos Business que guardan la información a persistir.
- ⑩ Observa el código de las clases DAO. Utilizan una clase llamada **db** que, básicamente, proporciona un método para conectarse a la BDD y otro para realizar consultas.
- ⑩ La clase db coge los parámetros de conexión a la BDD del fichero de configuración **config.inc.php**, donde están definidos como variables de ámbito global con \$GLOBALS
- ⑩ Pasos a seguir:
 - 1) Base de datos.
 - ⑨ Dentro del proyecto Pt1_version_0, hay un script para crear la BDD **bank**.
 - ⑨ Para ejecutar el script, podéis usar el cliente de BDD **phpmyadmin** que instalamos en la UD1. Escribe en la barra de direcciones del navegador:

localhost/phpmyadmin

- ⑨ Nueva → crea la BDD «**bank**». Es recomendable escoger un formato Unicode para evitar problemas con acentos o otros caracteres. Por ejemplo:

utf8mb4_unicode_ci



- ⑨ Una vez creada la BDD bank, en la sección SQL podemos copiar el código del script para generar la estructura de la BDD y cargar datos de muestra.
- 2) Prueba de funcionamiento.
 - ⑨ Cuando ejecutamos por primera vez el proyecto, nos encontramos que no funciona bien, y si debugamos marca un error de servidor.

- Este error es debido a que no se encuentra una de las clases de la subcapa *model/persistence*.
 - Lee atentamente el mensaje de error, identifica la clase que falta y hazla visible con **include** o **require_once** donde sea necesario.
- ⑨ Una vez resuelto el problema anterior, obtenemos un nuevo error, debido a que hay que adaptar los parámetros de conexión del fichero **config.inc.php** a la configuración de tu servidor MySQL.
- El servidor se encuentra en localhost.
 - El usuario root y su contraseña se configuró al instalar MySQL (si es Xampp, usuario root sin contraseña, ponemos como password cadena vacía).
 - A partir de aquí el proyecto ya funciona. Puedes probar de ver todas las cuentas o insertar una cuenta nueva.
 - Observa que la id de la cuenta está definida en la BDD como A_I (auto_increment).
 - En los apartados siguientes se piden varias modificaciones y mejoras.

3) Editar cuenta.

Se pide añadir la funcionalidad para editar una cuenta.

- ⑨ Añade al listado de cuentas un link para editar cada cuenta. Al clicar este link, se redirecciona a un controlador que tendrás que crear: **edit_cuenta_ctl.php**. Para enviar la id de la cuenta que hemos clicado a este controlador, puedes hacerlo como se hace al eliminar una cuenta.
- ⑨ Hay que crear un formulario de edición **form_edit_cuenta.php** que enviará al controlador de edición los valores introducidos por el usuario.
- ⑨ Puedes coger como modelo el formulario de inserción. Hay que poner los datos actuales de la cuenta como valores por defecto (atributo *value* de los inputs HTML).
- ⑨ También tendrás que añadir al DAO un método modificar().

← → ↻   localhost/DWES/ud6_pt1/controller/edit_cuenta_ctl.php?id=1

Edición de BK 100

Id	1
Código	BK 100
Saldo	15000
Cliente	1

[Volver](#)

- ⑨ Tal y como se ve en la captura, en el formulario de edición tenemos que enviar al controlador la id de la cuenta que estamos editando. 2 maneras de hacerlo:
 - ⑩ Añadir un input para la id, con el atributo *readonly* de valor true (para que el usuario no pueda modificar la id de la cuenta).
 - ⑩ Añadir un input de tipo *hidden* para enviar la id.

4) Funcionalidades del cliente.

Por último, se pide reproducir para el cliente las mismas funcionalidades que tenemos ahora para la cuenta. Habrá que crear la clase Business y el DAO para el cliente.

- ⑨ Tal y como vemos en la BDD, la clase *cliente* tendrá propiedades *id*, *dni*, *nombre*, *apellidos* y *fechaN* (fecha de nacimiento).

Observación: en los formularios para insertar o modificar un cliente, la fecha de nacimiento debería ser un input de tipo **date** en lugar de text como el resto de inputs.

- ⑨ Por otra parte, puedes comprobar que si insertamos un cliente lo hace bien, pero si intentamos insertar otro sale un mensaje de error, que habrá que solucionar.

Ayuda: el problema es que el campo *id* de la tabla *cliente* de la BDD *bank* no es Autoincremental. Tenemos que hacer que lo sea. Puedes conseguirlo con phpMyAdmin.

La aplicación debería quedar de la siguiente manera:

- ⑨ Página de inicio con dos links: «Menú Cuenta» y «Menú Cliente».
- ⑨ En cada una de estas dos páginas, un link para ver todos y otro para insertar una nueva cuenta o cliente.
- ⑨ En el listado de cuentas o clientes, habrá un link para modificar y otro para eliminar cada cuenta o cliente.
- ⑨ En la versión inicial de la aplicación, todos los links «Volver» van a la página de inicio.
 - ⑩ Ahora sólo lo haremos así desde el menú Cuenta y el menú Cliente.
 - ⑩ Tanto en el listado como en los formularios de la cuenta, el link «Volver» irá al menú de la cuenta.
 - ⑩ Tanto en el listado como en los formularios del cliente, el link «Volver» irá al menú del cliente.

Entrega de la práctica: carpeta comprimida con los ficheros de código fuente.