Práctica 2. Actividad Evaluable

PRÁCTICA

Desarrollo web en entorno servidor
2do CFGS DAW

Autores: Vanessa Tarí Costa - vanessa.tari@ceedcv.es
2020/2021

Licencia

Nomenclatura

A lo largo de este tema se utilizarán distintos símbolos para distinguir elementos importantes dentro del contenido. Estos símbolos son:

ÍNDICE

1. INTRODUCCIÓN

En esta actividad evaluable se van a repasar todos los conceptos vistos en las unidades anteriores añadiendo los conceptos de sesiones y cookies, conexión a bases de datos y objetos.

1.1 Competencias específicas

El alumnado será capaz de:

- Trabajar con un lenguaje de servidor como PHP
- Organizar el código en bibliotecas
- Utilizar funciones predefinidas en PHP
- Crear sus propias funciones en PHP
- Crear una base de datos en MySQL y conectar la aplicación
- Gestionar las cookies y sesiones dentro de la aplicación
- Crear objetos dentro de la aplicación
- Utilizar herramientas de desarrollo web

1.2 Competencias transversales

El alumnado será capaz de:

- Comunicar de forma escrita en el ámbito académico
- Adaptarse a las tecnologías y entornos de desarrollo para la creación de páginas web

1.3 Objetivos

Los objetivos que se desean alcanzar con esta práctica son:

- Gestionar cookies y sesiones
- Desarrollar una aplicación basada en la programación orientada a objetos
- Utilizar bases de datos para la recuperación y almacenamiento de la información

2. ENUNCIADO

Esta práctica es la continuación de la primera, en ella vamos a hacer modificaciones:

- La base de datos ya no se encontrará en ficheros, sino que se deberá crear en la base de datos de MySQL, ya sea por línea de comandos o a través de PhpMyAdmin.
- Se crearán CRUDS, para realizar operaciones sobre la base de datos.
- Se crearán clases para cada uno de los elementos que componen la aplicación: usuarios, películas, actores y directores

Para el diseño de la aplicación web yo he utilizado Bootstrap, no es necesario que lo utilicéis si no sabéis, pero el aspecto que hagáis debe ser similar.

En cuanto a la base de datos os tenéis que crear una llamada **videoclub** y que contenga las siguientes tablas:

TABLA USUARIOS

	NOMBRE	TIPO
CLAVE PRIMARIA	id	INT(11)
UNIQUE	email	VARCHAR(50)
	password	VARCHAR(50)
	guardaCredenciales	BOOLEAN

TABLA PELICULAS

	NOMBRE	TIPO
CLAVE PRIMARIA	id	INT(11)
UNIQUE	titulo	VARCHAR(50)
	anyo	VARCHAR(50)
	duracion	BOOLEAN

TABLA ACTORES

	NOMBRE	TIPO
CLAVE PRIMARIA	id	INT(11)
	nombre	VARCHAR(50)
	anyoNacimiento	VARCHAR(4)
	Pais	VARCHAR(50)

TABLA DIRECTORES

	NOMBRE	TIPO
CLAVE PRIMARIA	id	INT(11)
	nombre	VARCHAR(50)
	anyoNacimiento	VARCHAR(4)
	Pais	VARCHAR(50)

TABLA PELICULAS_ACTORES

	NOMBRE	TIPO
	id_pelicula	INT(11)
	Id_actor	INT(11)

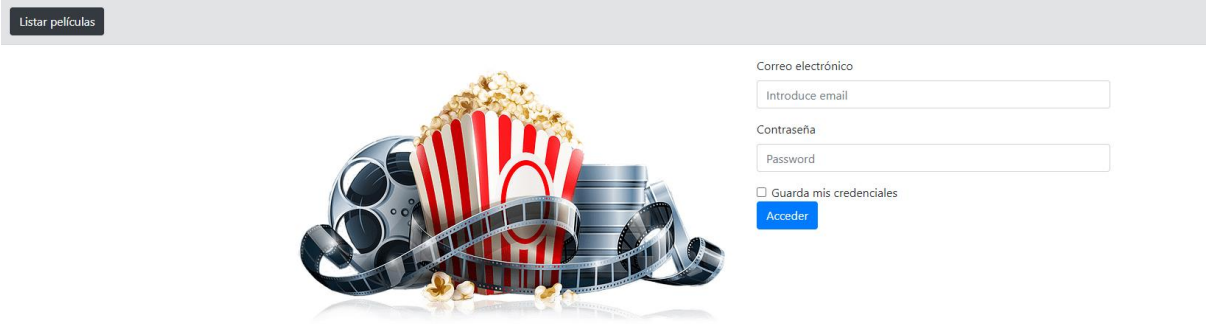
TABLA PELICULAS_DIRECTORES

	NOMBRE	TIPO
	id_pelicula	INT(11)
	id_director	INT(11)

Los datos los podréis importar de los CSV que se entregan con la práctica.

Página de index.php

En esta práctica la página de index.html mostrará un formulario de login. El usuario introducirá su email y contraseña, además podrá solicitar si desea que sus datos sean recordados.



Cuando el usuario introduce sus datos se debe comprobar que exista en la base de datos, por lo tanto, se debe almacenar el email y la contraseña en un objeto tipo Usuario que se pasará como parámetro al CRUD correspondiente de usuario.

Si el usuario es correcto se pasará la pantalla de películas.php, en caso contrario permanecerá en la misma y mostrará un error de credenciales erróneas.

Una vez el usuario ha entrado a la aplicación se creará la sesión para que se mantenga durante la navegación de todas las páginas. No se podrá acceder a la aplicación si el usuario no se ha autenticado, es por ello por lo que se deberá comprobar en todas las páginas si existe la sesión y sino devolver al usuario a la página de index.php.

Además, si el usuario ha marcado la opción de guardar credenciales, se almacenará en una cookie el id del usuario, que nos servirá para obtener sus datos en próximas visitas a la web.

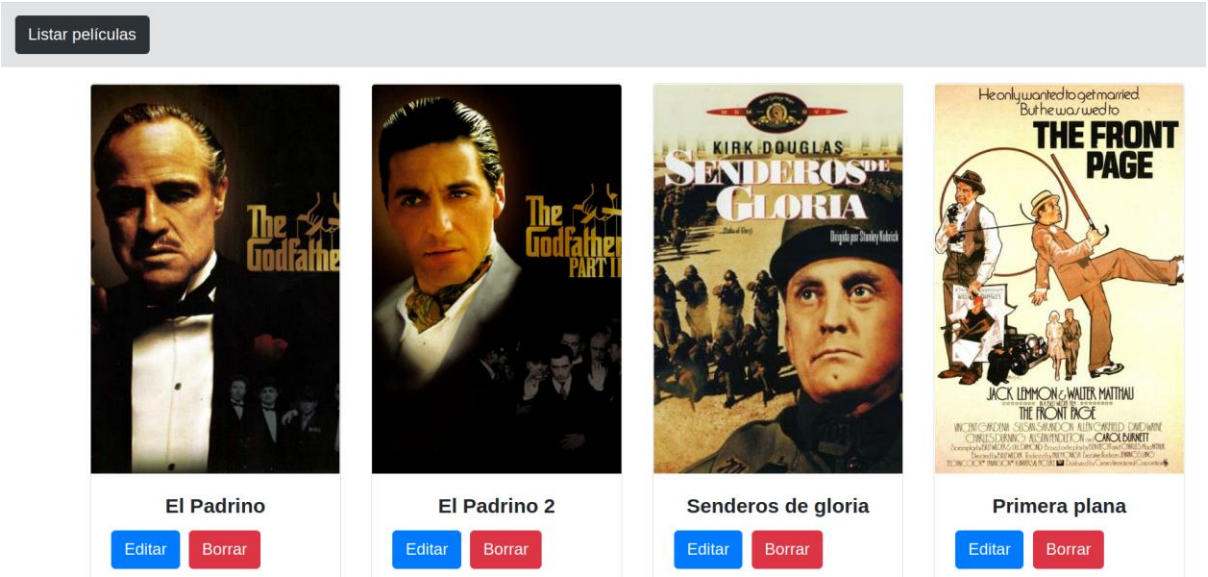
Página de películas.php

La página de películas se mostrará igual que en la práctica anterior, con la diferencia que a la hora de mostrar los datos estos deben ser recogidos de la base de datos en lugar del fichero. Para ello será necesario crear una clase Película, Actor y Director con sus CRUDS correspondientes.

Al pinchar en la imagen esta debe ser un enlace donde se pase por GET el id de la película y nos lleve a la página peliculas_ficha.php.

El botón de editar nos llevará a la página de películas_form.php, este botón será un enlace donde se pase por GET el id de la película.

Por el contrario, el botón de borrar se quedará en la misma página y borrará la película en la base de datos usando el CRUD correspondiente. En caso de que todo hay ido bien mostrará el mensaje “La película ha sido borrada correctamente”, en caso de error mostrará el mensaje “Error, ha habido un problema al borrar la película. Inténtelo de nuevo más tarde”.

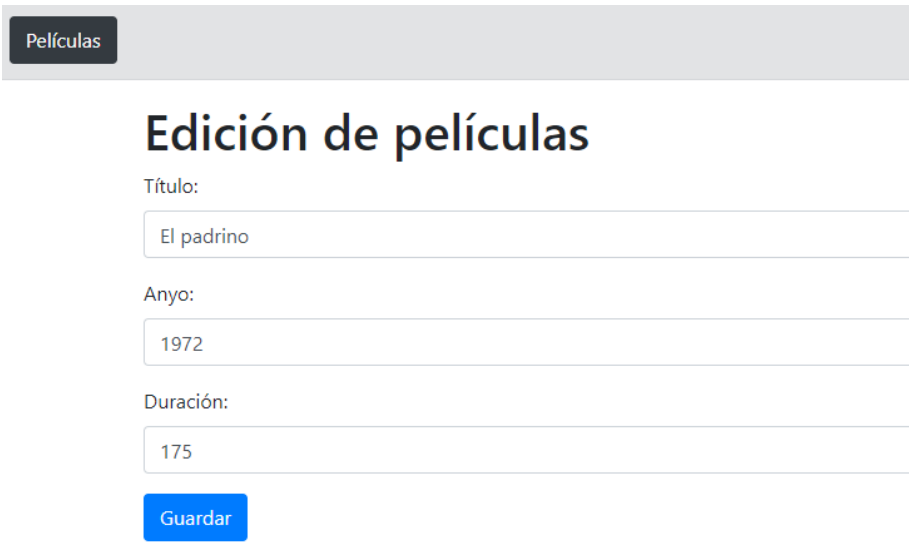


Página de películas_form.php

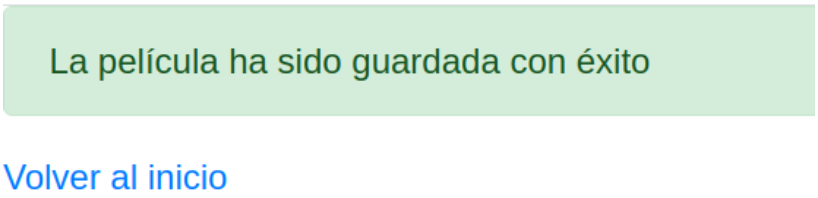
Esta página recibirá por GET el id de la película y deberá buscar los datos en la base de datos, haciendo uso de CRUD correspondiente.

Al igual que en la práctica anterior vamos a tener un formulario de edición de películas. Estos inputs deben rellenarse con los datos obtenidos desde la base de datos.

El botón guardar almacenará los datos en la tabla películas de la base de datos. Al darle a guardar se quedará en la misma página y guardará lo datos en la base de datos haciendo uso de la clase Película y su CRUD correspondiente.



En caso de que todo hay ido bien mostrará el mensaje “La película ha sido guardada correctamente”, en caso de error mostrará el mensaje “Error, ha habido un problema al guardar la película. Inténtelo de nuevo más tarde”.



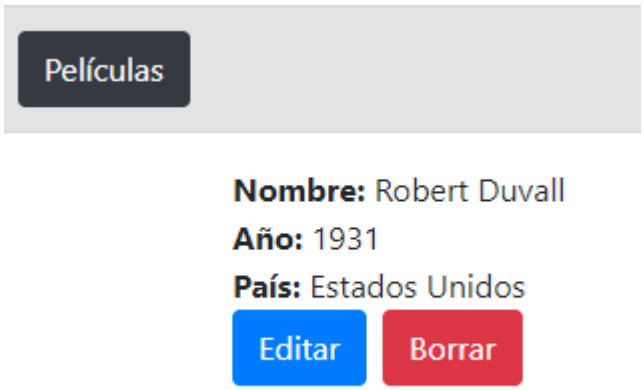
Página de peliculas_ficha.php

Esta página mostrará los datos de la película, así como los directores y actores de la misma. En esta página deberemos leer el id de la película pasado por GET y obtener los datos de la base de datos para ello usaremos la clase Película, Actor y Director y sus CRUDs correspondientes.

Al pinchar sobre el nombre del actor o director nos llevará la página actores_ficha.php y directores_ficha.php respectivamente. A este enlace se le pasará el id del actor si se trata de un actor o el id del director en caso de que sea un director.

Páginas actores_ficha.php y directores_ficha.php

En estas páginas recogeremos el id del actor o director y buscaremos en la base de datos. Para ello necesitaremos la clase Actor y Director y sus CRUDS correspondientes.



El botón de editar nos llevará al fichero actores_form.php o directores_form.php. Este botón será un enlace donde se pasará por GET el id del actor o el director. El botón borrar nos llevará a la página actores_borrado.php o directores_borrado.php que se encargarán de borrar de la base de datos el actor o director correspondiente.

Los ficheros actores_form.php y directores_form contendrán un formulario donde deben aparecer en los inputs los datos recogidos de la base de datos.

Por el contrario, el botón de borrar se quedará en la misma página y borrará el actor o director en la base de datos usando el CRUD correspondiente. En caso de que todo hay ido bien mostrará el mensaje “El actor ha sido borrado correctamente”, en caso de error mostrará el mensaje “Error, ha habido un problema al borrar el actor. Inténtelo de nuevo más tarde”.

3. PÁGINA DE DATABASE.PHP

Será imprescindible crear una clase Database que se encargue de gestionar la conexión y desconexión a la base de datos. Esta conexión será una propiedad estática de la clase Database.

4. CRUDS

Todos los CRUDS creados deberán tener al menos los siguientes métodos que nos servirán para futuras ampliaciones del proyecto:

- insertar($pelicula), el parámetro será de tipo Película, Actor, Director o Usuario, dependiendo del CRUD en cuestión
- mostrar(), esta función devolverá una lista de objetos de tipo Pelicula, Actor, Director o Usuario, dependiendo del CRUD en cuestión
- eliminar($id), esta función buscará el dato por el id pasado y lo eliminará de la base de datos
- obtener($id), esta función devolverá un único objeto del tipo que corresponda en cada CRUD.
-actualizar($pelicula), está función actualizará la base de datos con los datos obtenidos por parámetro. En el caso de actor, director o usuario el método recibirá por parámetro su tipo correspondiente.

Además, el CRUD de peliculas contendrá dos método más:

- obtenerActoresPelicula($id), esta función devuelve los actores relacionados con la película donde el parámetro que se le pasa es el id de la película
- obtenerDirectoresPelicula($id), esta función devuelve los directores relacionados con la película donde el parámetro que se le pasa es el id de la película

5. PLAGIO

Las actividades evaluables deben realizarse de forma individual, en caso de plagio, **la nota del módulo será de 0.**

6. FORMATO Y FECHA DE ENTREGA

Para la entrega se utilizará un repositorio Git. Al finalizar el entregable se debe enviar mediante una “tarea” en nuestro Moodle, un enlace al repositorio GitHub privado con la práctica realizada y permiso para la cuenta de GitHub vtari.

Para instalar Git:

- Ubuntu:
  - sudo apt-get update
  - sudo apt-get install git
- Windows: https://git-for-windows.github.io/

Para facilitar la tarea del uso de Git es recomendable instalar alguna extensión o entorno que os facilite su uso.

Para usar Git en Visual Studio Code recomendamos:

- https://code.visualstudio.com/docs/editor/versioncontrol
- http://www.mclibre.org/consultar/informatica/lecciones/vsc-git-repositorio.html

Aquí un ejemplo del uso de Git dentro de Visual Studio Code:
https://code.visualstudio.com/docs/introvideos/versioncontrol

FECHA LÍMITE DE ENTREGA: 24 de enero a las 23:59

7. MATERIAL ADICIONAL

[1] https://www.php.net/manual/es/index.php
[2] https://getbootstrap.com/docs/4.5/getting-started/introduction/
[3] https://code.visualstudio.com/docs/introvideos/versioncontrol

8. AUTORES

A continuación, ofrecemos en orden alfabético el listado de autores que han hecho aportaciones a este documento:

- Vanessa Tarí Costa