



Proyecto Watched!

ASIGNATURA: APLICACIONES WEB

AN WEI PHAM LUO

ALEJANDRO POVEDANO ATIENZA

GONZALO FERNÁNDEZ-DÍEZ PONTE

RAQUEL RAMOS CORRAL

VÍCTOR GÓMEZ-JAREÑO GUERRERO

VÍCTOR RAMOS FUENTES

Índice

Contenido

<i>INTRODUCCIÓN</i>	2
<i>DESCRIPCIÓN DETALLADA DE LA APLICACIÓN</i>	3
<i>ARQUITECTURA DE LA APLICACIÓN</i>	11
Listado de Scripts de Lógica de Negocio	11
Listado de Scripts Adicionales	13
Estructura de la base de datos	14
<i>INSTRUCCIONES DE INSTALACIÓN</i>	20

INTRODUCCIÓN

Watched es una aplicación que nació en un laboratorio de la Facultad de Informática de la Universidad Complutense de Madrid. Esta aplicación está pensada principalmente como un gestor de películas, la cual permite hacer un seguimiento de sus visualizaciones.

¿Qué hace especial a esta aplicación con respecto a lo que ya hay en el mercado?

Fácil, Watched dispone de unos retos que el usuario podrá ir completando. Dichos retos están actualizados en las novedades de la industria. También habrá retos con la temática propia de premios cinematográficos como pueden ser los Goya o los Óscars.

Los usuarios dispondrán de la opción “Crear Listas” donde podrán incluir películas de la misma temática. Por ejemplo, listas de películas de DC o del género comedia.

Finalmente, la aplicación facilitará una serie de rankings donde se podrá consultar quienes han visualizado el mayor número de películas, los usuarios que más listas han creado, etc...

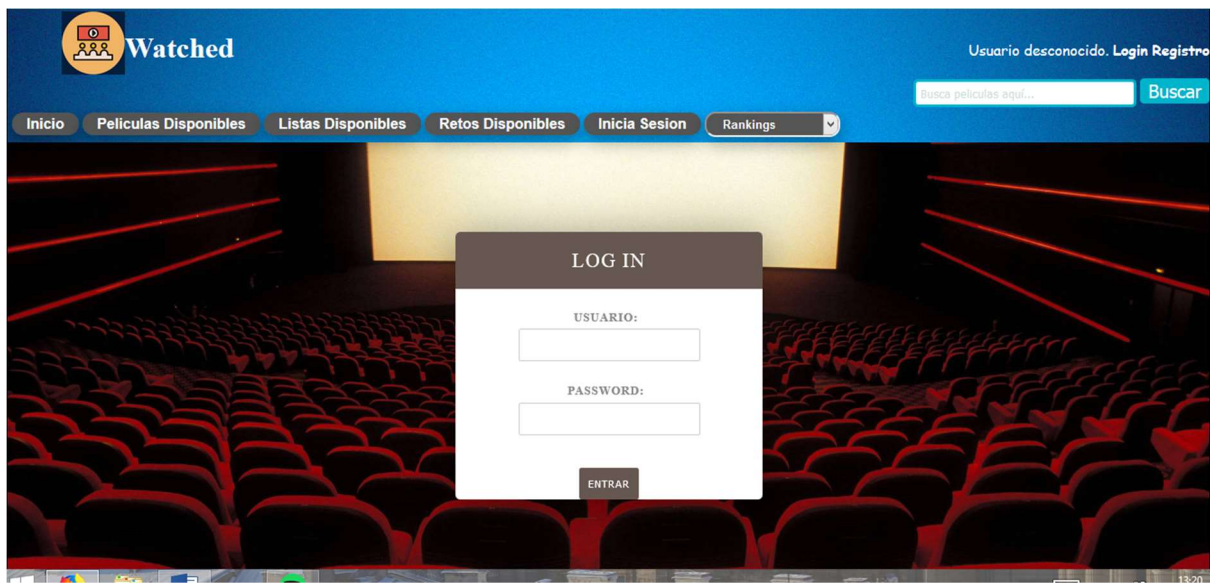
DESCRIPCIÓN DETALLADA DE LA APLICACIÓN

Sitio Web: container.fdi.ucm.es:20101

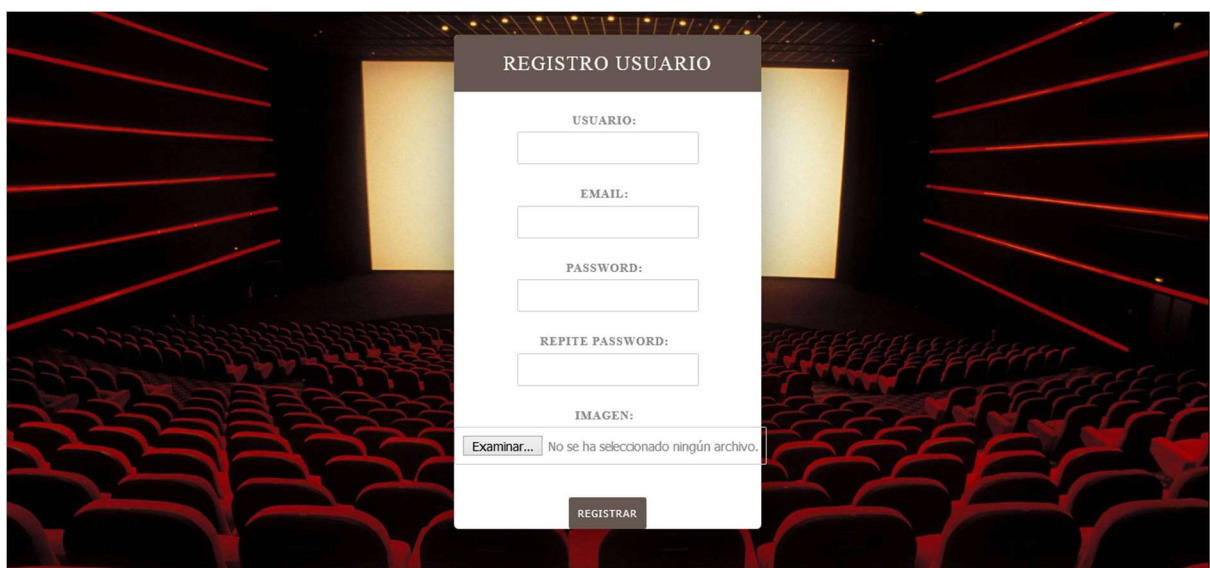


index.php: este fichero genera la página principal de la aplicación, es decir, aquella que solicita el navegador al entrar en la URL de la web.

Desde aquí podemos acceder mediante los botones de la cabecera al listado de películas disponibles, rankings, retos e incluso iniciar sesión o registrarse. Todas estas funcionalidades serán detalladas a continuación.



[login.php](#): en este fichero se genera la vista de los campos a introducir para que un usuario pueda loguearse, es decir, nombre y contraseña. Se verifican con la Base de Datos para comprobar su validez.



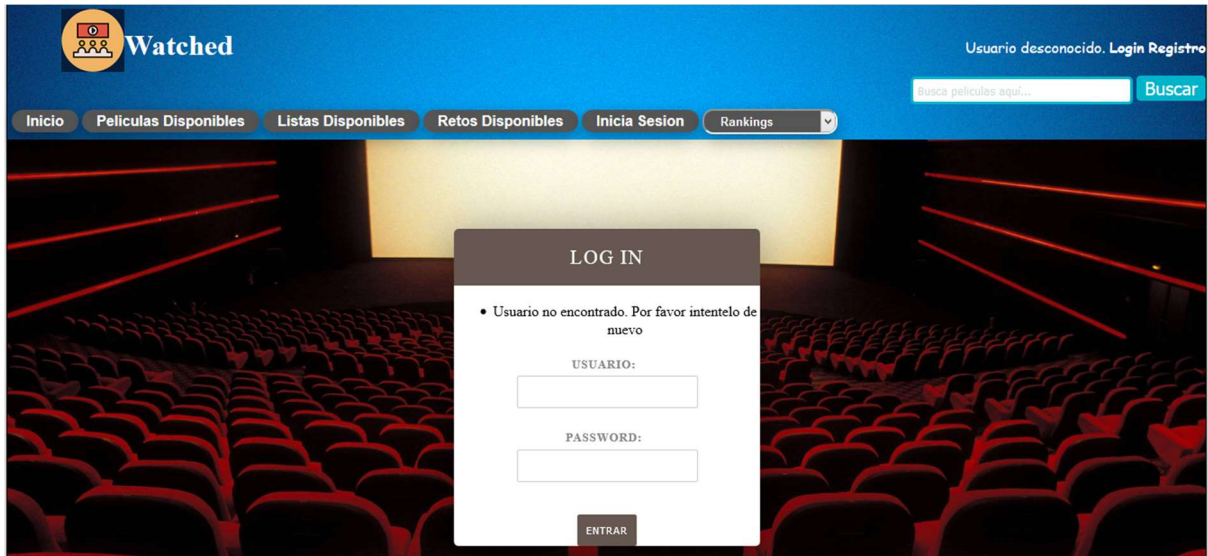
[registro.php](#): al igual que login.php, genera los campos necesarios para que el usuario se registre. El nombre de usuario no se puede repetir con uno ya existente, al igual que el nombre de la imagen no puede ser el mismo que otro. Si no insertas una imagen, se asignará una por defecto.

Ahora vamos a simular que un usuario (cuyo nombre sea Victor, por ejemplo) inicia sesión e introduce sus datos correctamente.

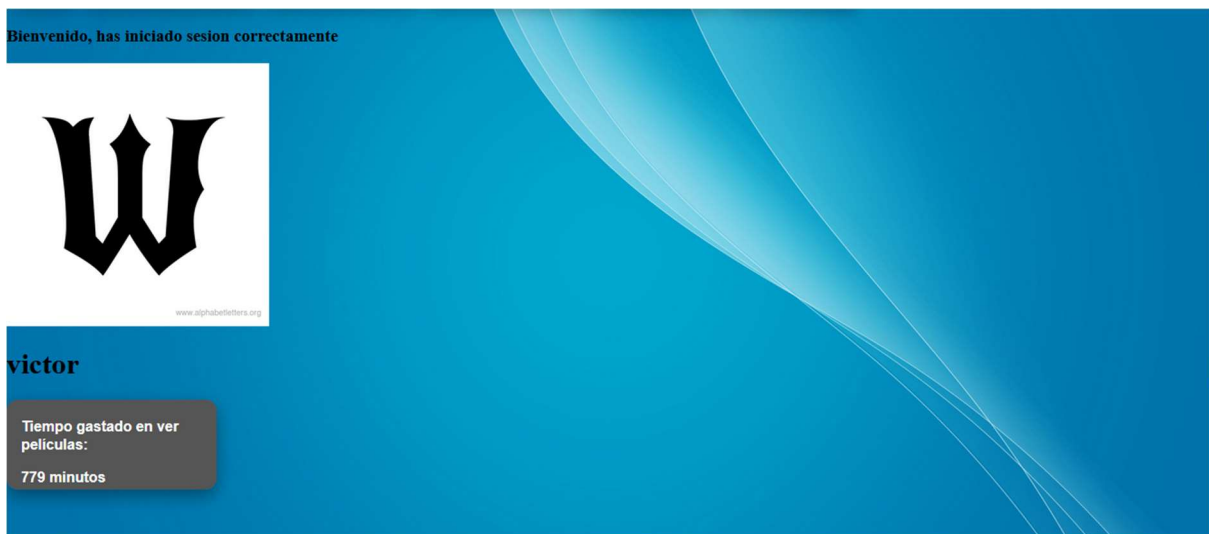


Vemos como en el margen superior derecho aparece el nombre del usuario con la opción de Salir (Cerrar Sesión).

Si por el contrario, introduce unos datos que no están incluidos en la base de datos, se le mostrará lo siguiente:



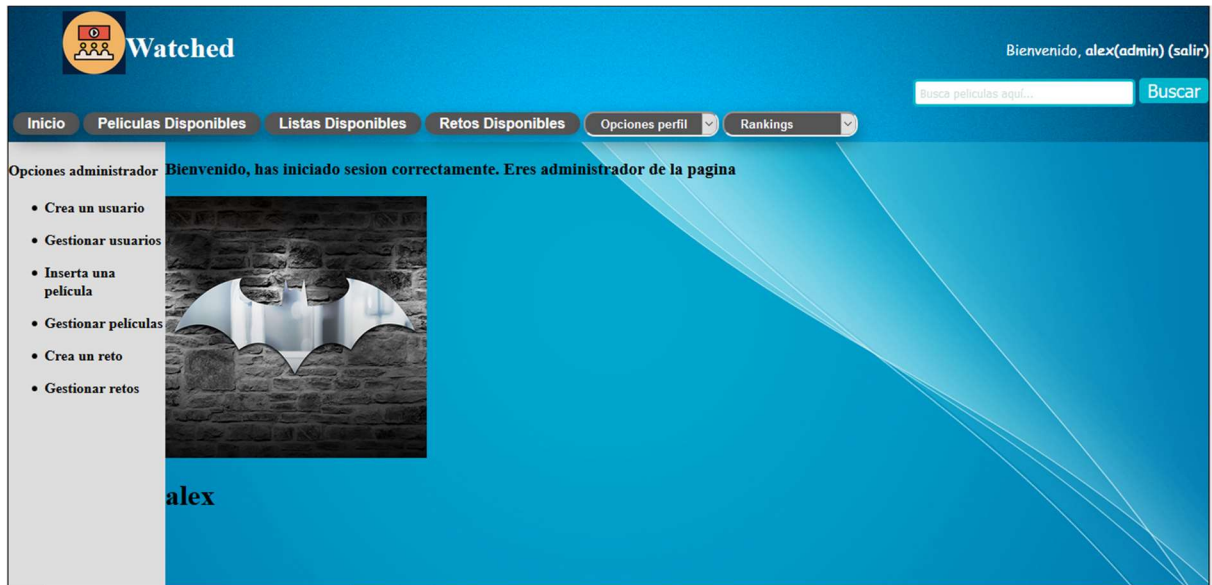
Una vez haya iniciado la sesión el usuario, se le mostrará su perfil.



El perfil contiene una imagen, el nombre del usuario y un contador con el número de minutos empleados viendo películas.

[perfil.php](#): la vista facilitada para un usuario sin privilegios, es decir, sin ser de tipo administrador, posee una serie de funcionalidades muy limitadas con respecto a las redactadas anteriormente, ya que únicamente se tendrá acceso a ver las películas marcadas como vistas, a crear listas de películas, a editar tu perfil, a ver tus listas seguidas y a ver tus listas creadas.

Ahora veremos qué sucede si el usuario que inicia sesión es el administrador de la página.



[perfilAdmin.php](#): aquí se genera la vista que se observa al loguearse como usuario de tipo administrador, este tipo de usuario puede realizar modificaciones en el resto de usuarios como en la lista de películas que dispone la web.

La cabecera se muestra igual, sin embargo se puede observar que el admin en vez de tener un contador con el número de películas vistas, posee en el margen izquierdo un listado de acciones que puede desempeñar.

Una vez detallado el proceso de login y registro, vamos a ver cómo funciona la gestión de películas.


Si pulsamos sobre el botón de películas disponibles, se mostrará lo siguiente.



[listaPelículas.php](#): en este php se genera la vista de todas y cada una de las películas que el usuario puede ver o incluir en alguna de sus listas. Igualmente, si no hay un usuario

registrado, éste también podrá acceder al listado al completo pero sin poder interaccionar con el botón vista o la funcionalidad de añadir la película a una lista.

Al clicar sobre la imagen de una película se mostrará la información referente a esta.




Genero: Acción.

Año: 2005

Duracion: 140 minutos.

Actores:

- Christian Bale
- Michael Caine

 (No Vista)

Agregar a Lista:

Batman Begins

Sinopsis: Nueva adaptación del famoso cómic. Narra los orígenes de la leyenda de Batman y los motivos que lo convirtieron en el representante del Bien en la ciudad de Gotham. Bruce Wayne vive obsesionado con el recuerdo de sus padres, muertos a tiros en su presencia. Atormentado por el dolor, se va de Gotham y recorre el mundo hasta que encuentra a un extraño personaje que lo adiestra en todas las disciplinas físicas y mentales que le servirán para combatir el Mal. Por esta razón, la Liga de las Sombras, una poderosa y subversiva sociedad secreta, dirigida por el enigmático Ra's Al Ghul, intenta reclutarlo. Cuando Bruce vuelve a Gotham, la ciudad está dominada por el crimen y la corrupción. Con la ayuda de su leal mayordomo Alfred, del detective de la policía Jim Gordon y de Lucius Fox, su colega en una Sociedad de Ciencias Aplicadas, Wayne libera a su alter ego: Batman, un justiciero enmascarado que utiliza la fuerza, la inteligencia y la más alta tecnología para luchar contra las siniestras fuerzas que amenazan con destruir la ciudad.

Comentarios:

Aun no has comentado esta película.

Comentarios:


Valoracion:

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

[película.php](#): aquí se detallan todos los campos de información a los que podemos acceder sobre una película, tales como: título, género, duración, año, portada, actores que participan en la misma y sinopsis.

En el panel izquierdo se detallan datos característicos de la película como actores o género entre otros y se facilita un botón para marcar la película como vista, bien sea para ampliar nuestro contador de minutos mostrado en el perfil, ya mencionado anteriormente, o bien para completar alguno de los retos (esta parte se explicará más adelante).

También se dará la posibilidad de agregar la película a una lista, en el caso de que el usuario esté creando una lista de películas protagonizadas por Christian Bale esta debe estar incluida.




Genero: Acción.

Año: 2005

Duracion: 140 minutos.

Actores:

- Christian Bale
- Michael Caine

 (Vista)

Agregar a Lista:

Batman Begins

Sinopsis: Nueva adaptación del famoso cómic. Narra los orígenes de la leyenda de Batman y los motivos que lo convirtieron en el representante del Bien en la ciudad de Gotham. Bruce Wayne vive obsesionado con el recuerdo de sus padres, muertos a tiros en su presencia. Atormentado por el dolor, se va de Gotham y recorre el mundo hasta que encuentra a un extraño personaje que lo adiestra en todas las disciplinas físicas y mentales que le servirán para combatir el Mal. Por esta razón, la Liga de las Sombras, una poderosa y subversiva sociedad secreta, dirigida por el enigmático Ra's Al Ghul, intenta reclutarlo. Cuando Bruce vuelve a Gotham, la ciudad está dominada por el crimen y la corrupción. Con la ayuda de su leal mayordomo Alfred, del detective de la policía Jim Gordon y de Lucius Fox, su colega en una Sociedad de Ciencias Aplicadas, Wayne libera a su alter ego: Batman, un justiciero enmascarado que utiliza la fuerza, la inteligencia y la más alta tecnología para luchar contra las siniestras fuerzas que amenazan con destruir la ciudad.

Comentarios:

Aun no has comentado esta película.

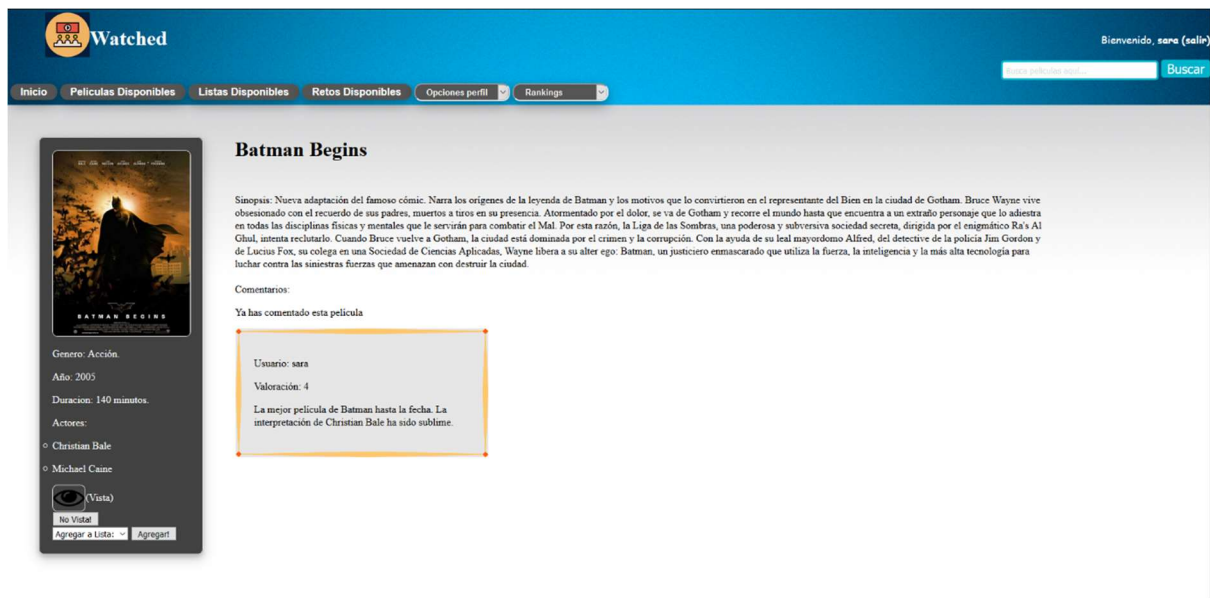
Comentarios:

Valoracion:

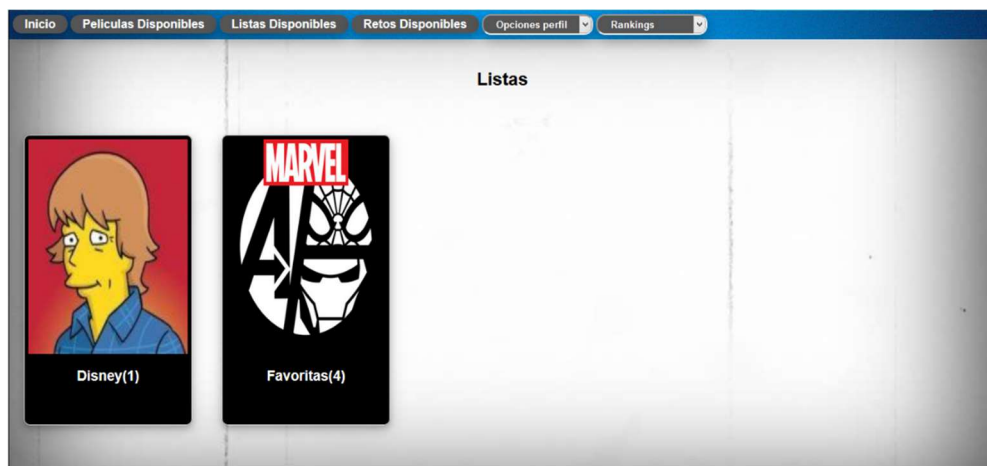
☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

Si el usuario pulsa el botón de vista, se le reconducirá a la misma página indicando que esa película ya ha sido vista y dando la posibilidad al usuario de cancelar la acción realizada anteriormente en el caso de que se haya equivocado.

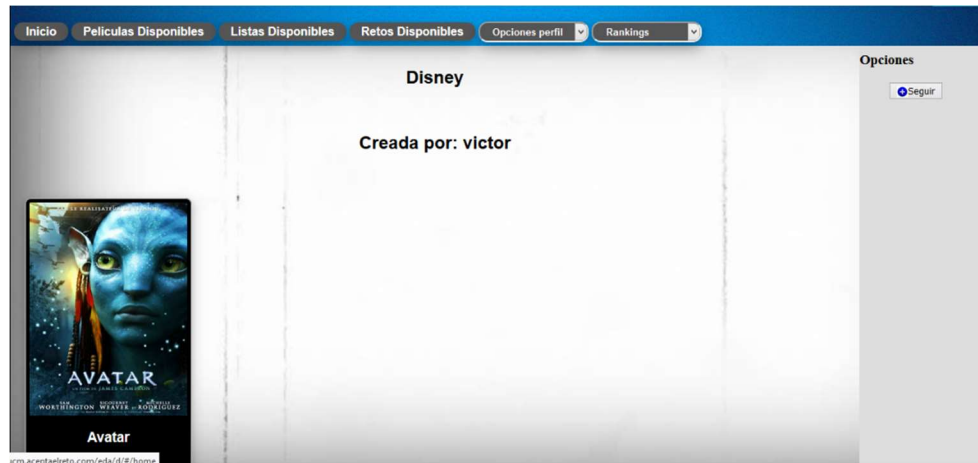
Debajo de la sinopsis, están implementados los comentarios. Dando la posibilidad a un usuario de añadir una crítica y una valoración sobre 5. Un usuario no registrado no puede ver los comentarios y se le invitará a registrarse.



Otra funcionalidad implementada son las listas de películas que han sido creadas por los usuarios, ya hemos hablado sobre ello anteriormente, pero ahora se añadirán unas imágenes para dejarlo más claro.

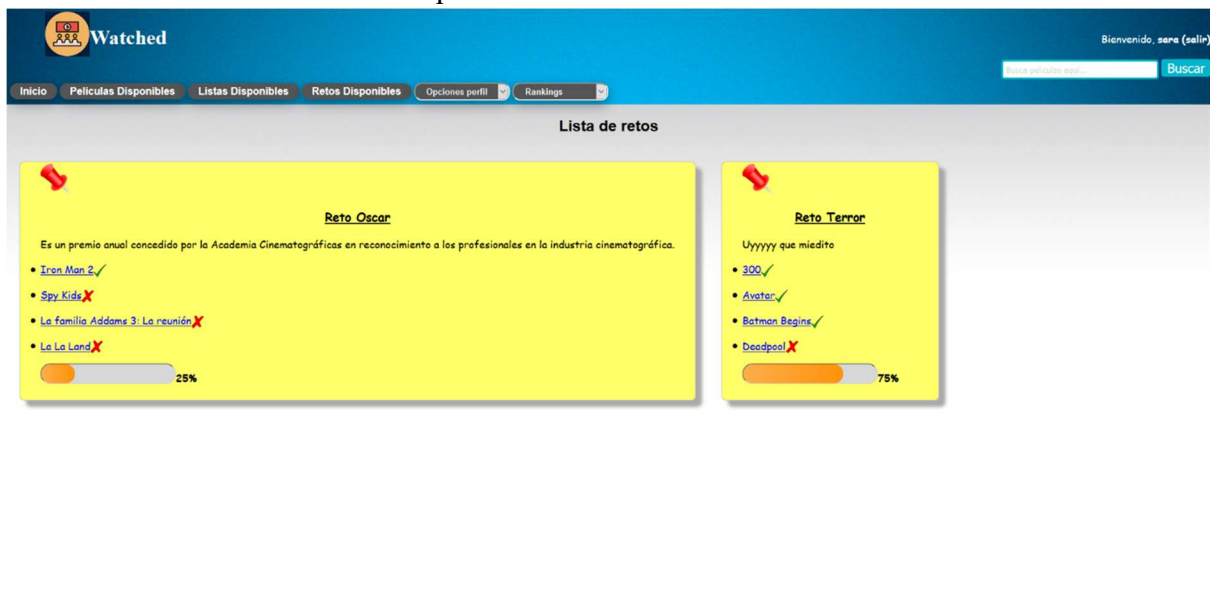


Dentro de una lista tienes dos opciones, dependiendo de si eres creador o ajeno. Si eres de esta última opción, puedes seguir la lista para tenerla guardada en tu perfil. Si eres creador, puedes eliminarla, cambiar su nombre o eliminar películas.



Al clicar sobre la imagen de la película, la web se redireccionará hacia la información de la película, algo sobre lo que ya se ha hablado.

Los retos son otra funcionalidad que se va a detallar a continuación.

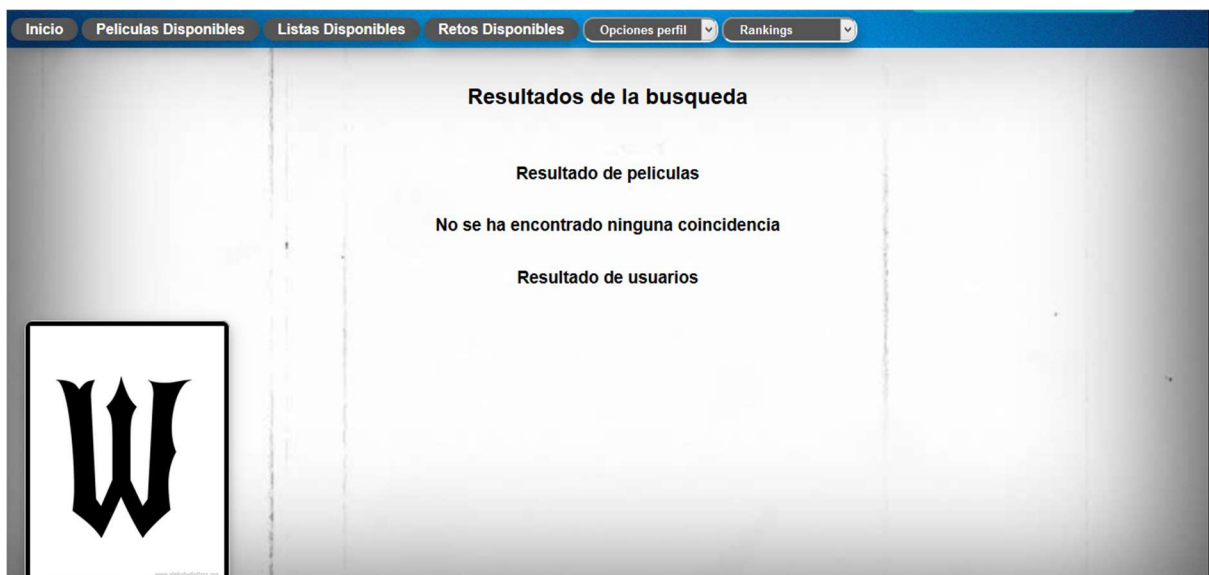


Se muestra por cada reto un listado de las películas pertenecientes al reto y una barra de progreso según los objetivos cumplidos. Aparte de una descripción para orientar sobre el objetivo del reto.

La aplicación también cuenta con una serie de rankings como el mostrado a continuación



Existen varios rankings de distinta modalidad como pueden ser los actores más veteranos o los usuarios con más listas...etc.



La barra de búsqueda situada en el margen superior derecho muestra los resultados en la lista de películas y en la lista de usuarios, en la imagen se ha buscado todos los objetos cuyo nombre contuviese “vi”.

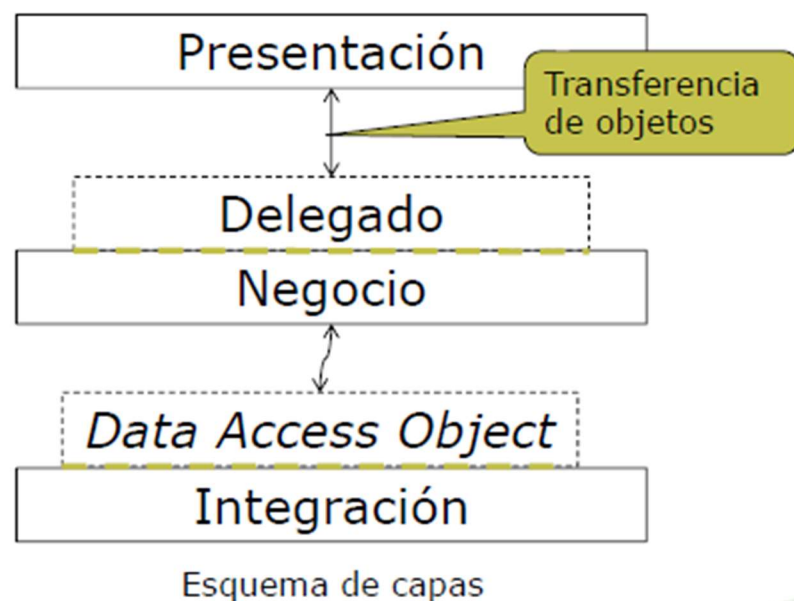
[vistaBusqueda.php](#): en este php se muestra la película o películas que contienen relación con la palabra introducida en la barra de búsqueda de la aplicación.

ARQUITECTURA DE LA APLICACIÓN

Listado de Scripts de Lógica de Negocio

Para el desarrollo de la parte lógica de la aplicación, nos hemos basado en un modelo de arquitectura de capas de sistemas de información, se distinguen tres tipos:

- Capa de Presentación: gestionan las interfaces con el usuario, son el listado de scripts detallados en el punto anterior.
- Capa de Negocio: implementan las reglas de gestión de datos y servicios de la aplicación, estos scripts se detallarán a continuación.
- Capa de Integración: Comunican a los otros componentes con recursos y sistemas externos, también se detallarán a continuación.



La capa de negocio está formada por dos tipos de patrones de diseño software, que son:

- Application Service (AS) : permite centralizar la lógica de negocio. Los scripts utilizados con este patrón son:
 - AS_listaPeliculas.php: este fichero contiene la clase AS_ListaPeliculas con 19 funciones que gestionan la sección de listas de películas.
 - AS_Pelicula.php: el archivo contiene la clase AS_Pelicula con 8 funciones que gestionan la sección de películas.
 - AS_Usuario.php: contiene la clase AS_Usuario con 10 funciones que gestiona toda la sección de usuarios.
 - AS_PeliculaVista.php: contiene la clase AS_PeliculaVista con 5 funciones que gestionan la sección de películas vistas.
 - AS_Comentarios.php: contiene la clase AS_Comentarios con 4 funciones que gestionan la sección de comentarios.
 - AS_Reto.php: contiene la clase AS_Retos con 8 funciones que gestionan la sección de retos.
 - AS_Ranking.php: contiene la clase AS_Comentarios con 5 funciones que gestionan la sección de ranking.

- Transferencia (TRANSFER) : este patrón está desarrollado con la función de pasar datos con múltiples atributos de una vez desde el cliente al servidor o viceversa. Los ficheros que implementan este patrón son:
 - Transfer_Pelicula.php : contiene las funciones get y set de los atributos : código, título, portada, año, duración, género y sinopsis.
 - Transfer_Usuario.php : contiene las funciones get y set de los atributos : nombre, contraseña, email, contPelis y userType.
 - Transfer_ListaPeliculas.php : contiene las funciones get y set de los atributos : código, título y nombreCreador.
 - Transfer_PeliculaVista.php : contiene las funciones get y set de los atributos : código y nombreusu.
 - Transfer_Reto.php: contiene las funciones get y set de los atributos : nombre, número y descripción.
 - Transfer_Comentarios.php: contiene las funciones get y set de los atributos: código, película, usuario, valoración y comentario.

La capa de integración es responsable de la comunicación con recursos y sistemas externos como por ejemplo la BBDD, el patrón de diseño utilizado de esta capa es:

- Data Access Object (DAO) : permite acceder a la capa de datos, es decir, enlaza la aplicación con la base de datos correspondiente. Los scripts con este patrón son :
 - DAO_Usuario.php: la clase userDAO contiene las funciones de creación, modificación, actualización y eliminación de un usuario.
 - DAO_ListaPeliculas.php: la clase listasPeliculasDAO contiene las funciones de creación, modificación, actualización y eliminación de una lista de películas.

- [DAO_Pelicula.php](#): la clase peliculasDAO contiene las funciones de creación, modificación, actualización y eliminación de una película.
- [DAO_peliculaVista.php](#): la clase peliculaVistaDAO contiene las funciones de creación, modificación, actualización y eliminación de una película vista.
- [DAO_comentario.php](#): la clase comentarioDAO contiene las funciones de creación, modificación, actualización y eliminación de un comentario.
- [DAO_ranking.php](#): la clase rankingDAO contiene las funciones de creación, modificación, actualización y eliminación de un ranking.
- [DAO_retos.php](#): la clase retosDAO contiene las funciones de creación, modificación, actualización y eliminación de un reto.

Listado de Scripts Adicionales

- [Form.php](#): es una clase abstracta que crea la parte del formulario en html, al ser abstracta permite cambiar los campos del formulario, y lo procesa. Esta clase crea un formulario html y lo inserta en la clase desde el cual se le llama.

Las clases que heredan de Form:

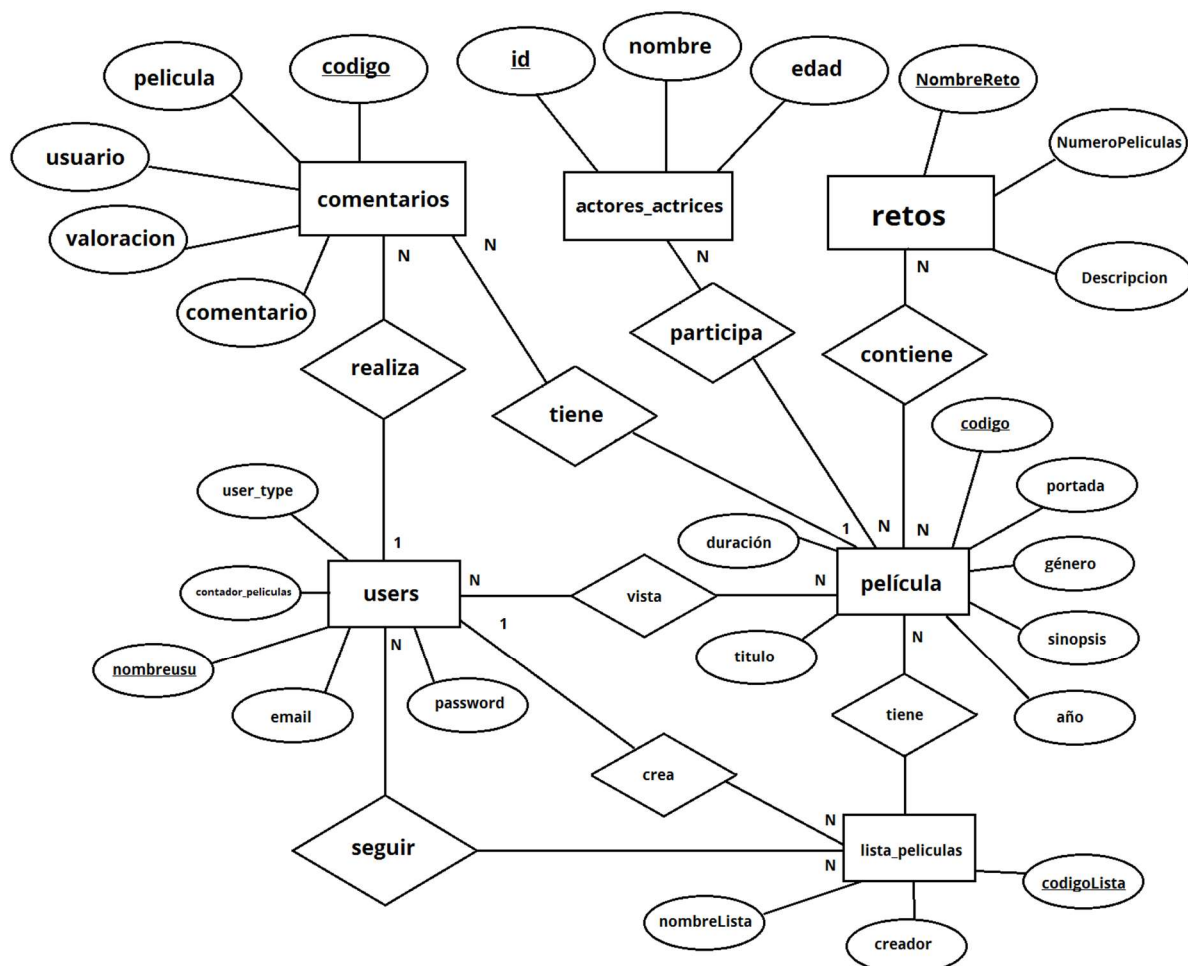
- [formListPelisUsu.php](#)
- [formLogin.php](#)
- [formRegistro.php](#)
- [formCreatePelicula.php](#)
- [formCreateReto.php](#)
- [formCreateUsuario.php](#)
- [formDeletePelicula.php](#)
- [formDeleteUsuario.php](#)
- [formUpdatePelicula.php](#)
- [formUpdatePerfil.php](#)
- [formUpdateReto.php](#)
- [formUpdateUsuario.php](#)

Son clases que se encargan establecer los campos del formulario para la creación de formularios específicos y de procesar los datos recogidos en los campos de manera diferente en cada uno de ellos.

- [conexion.php](#): este script contiene dos funciones (conectar y desconectar) que se encargan de hacer la conexión y la desconexión de la base de datos, siendo así un script que solo utilizan los DAO.

Estructura de la base de datos

La base de datos está originada en <https://www.phpmyadmin.net/>, una herramienta escrita en PHP cuya intención es facilitar la administración de MySQL a través de Internet.



Mapa de entidad-relación de nuestra BBDD

La tabla de **películas_vistas** de la Base de Datos no se ve reflejada en el mapa porque se genera a partir de la relación N:N entre **users** y **películas**.

La tabla de **listas_y_peliculas** de la Base de Datos no se ve reflejada en el mapa porque se genera a partir de la relación N:N entre **lista_peliculas** y **película**.

La tabla de **listas_y_usuarios** de la Base de Datos no se ve reflejada en el mapa porque se genera a partir de la relación N:N entre **lista_peliculas** y **usuarios**.

La tabla de **retos_y_peliculas** de la Base de Datos no se ve reflejada en el mapa porque se genera a partir de la relación N:N entre **reto** y **película**.

La tabla de **actores_y_peliculas** de la Base de Datos no se ve reflejada en el mapa porque se genera a partir de la relación N:N entre **actores_actrices** y **película**.

En nuestro proyecto la base de datos tiene por nombre “**bd_watched**” y posee diversas tablas que serán detalladas a continuación:

Users:

La finalidad de esta tabla es almacenar los distintos usuarios que se vayan registrando en nuestra página. Tiene una relación N:N con la tabla Película. Aquí se especifican los campos que contiene:

- “**nombreusu**”: nombre del usuario, el tamaño del campo son 20 bytes como máximo.
- “**email**”: e-mail del usuario, el tamaño del campo son 25 bytes como máximo.
- “**usertype**”: cuyo valor será 1 en caso de que el usuario sea administrador y 0 en caso contrario.
- “**password**”: contraseña del usuario, el tamaño del campo es de 25 bytes como máximo.
- “**Contador Peliculas**”: contador del número de películas vistas por el usuario, el tamaño del campo es de 11 bytes como máximo.
- “**fotoPerfil**”: El nombre de la foto que se almacena en una carpeta aparte, guardada en tipo text

Al crear un nuevo usuario, todos los campos deben tener asignado un valor, es decir, el usuario no conseguirá registrarse si deja algún campo en blanco. Excepto el valor de contador de películas que se pone por defecto a 0.

Los scripts que acceden a la base de datos de usuarios son:

- **conexion.php** : script que accede a la base de datos.
- **DAO_Usuario.php** : clase que contiene funciones que realizan modificaciones y/o consultas en la tabla **usuario**.

lista peliculas:

La finalidad de esta tabla es almacenar las distintas listas de películas que los usuarios vayan creando. Tiene una relación 1:N con la tabla Users. Aquí se especifican los campos que contiene:

- “**nombreLista**”: Título de la lista, máximo 535 bytes, sin posibilidad de no tener valor al crear una nueva entrada en la BD.
- “**CódigoLista**”: Código de la lista, máximo 255 bytes, sin posibilidad de no tener valor al crear una nueva entrada en la BD.
- “**creador**”: nombre del usuario que lo ha creado, el tamaño del campo son 20 bytes como máximo.
- “**numeroPeliculas**”: número total de películas que contiene esta lista.
- “**privacidad**”: 0 para lista pública y 1 para lista privada, dando acceso a los usuarios ajenos a esta lista o quitándoselo.

Los scripts que acceden a la base de datos de películas son:

- **conexion.php** : script que accede a la base de datos.

- **DAO_ListaPelículas.php** : clase que contiene funciones que realizan modificaciones y/o consultas en la tabla **lista_película**.

Película:

La finalidad de esta tabla es almacenar los distintas películas que tenemos disponibles, con la posibilidad de añadir más si eres administrador. Tiene una relación N:N con la tabla Users. Aquí se especifican los campos que contiene:

- **“Título”**: Título de la película, máximo 535 bytes, sin posibilidad de no tener valor al crear una nueva entrada en la BD.
- **“Código”**: Código de la película, máximo 255 bytes, sin posibilidad de no tener valor al crear una nueva entrada en la BD.
- **“Portada”**: Portada de la película guardada en un tipo text para luego acceder a ella en una carpeta.
- **“Genero”**: Género de la película en tipo text.
- **“Duracion”**: Duración de la película guardada en tipo entero. Se entiende que son los minutos totales que dura esa película.
- **“AñoEstreno”**: Año en el cual la película salió a los cines, guardada en tipo text.
- **“Sinopsis”**: Resumen que explica la trama de la película. Guardada en tipo text.

Los scripts que acceden a la base de datos de películas son:

- **conexion.php** : script que accede a la base de datos.
- **DAO_Película.php** : clase que contiene funciones que realizan modificaciones y/o consultas en la tabla **película**.

Películas vistas:

La finalidad de esta tabla es almacenar la relación entre Users y Película. Aquí se especifican los campos que contiene:

- **“Nombre Usuario”**: clave externa en relación con “**nombreusu**” de la tabla users.
- **“Código Películas”**: clave externa en relación con “**Código**” de la tabla películas.

Ningún valor podrá ser nulo al crear una nueva entrada en la tabla.

El script que accede a la base de datos de películas vistas es:

- **conexion.php** : script que accede a la base de datos.
- **DAO_PelículaVista.php** : clase que contiene funciones que realizan modificaciones y/o consultas en la tabla **película_vistas**.

listas y películas:

La finalidad de esta tabla es almacenar la relación entre **lista_peliculas** y **película**. Aquí se especifican los campos que contiene:

- **“listaPelículas”**: clave externa en relación con **“codigoLista”** de la tabla users.
- **“códigoPelícula”**: clave externa en relación con **“Código”** de la tabla lista_películas.

Ningún valor podrá ser nulo al crear una nueva entrada en la tabla.

El script que accede a la base de datos de listas_y_peliculas es:

- **conexion.php** : script que accede a la base de datos.
- **DAO_ListaPeliculas.php** : clase que contiene funciones que realizan modificaciones y/o consultas en la tabla **listas_y_peliculas**.

listas y usuarios:

La finalidad de esta tabla es almacenar la relación entre **lista_peliculas** y **users**. Aquí se especifican los campos que contiene:

- **“nombreUsu”**: clave externa en relación con **“nombreusu”** de la tabla users.
- **“listasPelis”**: clave externa en relación con **“codigoLista”** de la tabla lista_películas.

Ningún valor podrá ser nulo al crear una nueva entrada en la tabla.

El script que accede a la base de datos de listas_y_usuarios es:

- **conexion.php** : script que accede a la base de datos.
- **DAO_ListaPeliculas.php** : clase que contiene funciones que realizan modificaciones y/o consultas en la tabla **listas_y_usuarios**.

retos:

La finalidad de esta tabla es almacenar los retos de los que dispone la aplicación. Aquí se especifican los campos que contiene:

- **“NombreReto”**: Tipo text que contiene el nombre del reto.
- **“NumeroPeliculas”**: tipo entero que contiene un contador de las películas que tiene este reto.
- **“Descripcion”**: Un breve resumen de en que consiste el reto.

El script que accede a la base de datos de retos es:

- **conexion.php**: script que accede a la base de datos.
- **DAO_Reto.php**: clase que contiene funciones que realizan modificaciones y/o consultas en la tabla **retos**.

retos y películas:

La finalidad de esta tabla es almacenar la relación N:N entre retos y películas. Aquí se especifican los campos que contiene:

-“**NombreReto**”: Tipo text que contiene el nombre del reto.

-“**CódigoPelícula**”: clave externa en relación con “**CódigoPelícula**” de la tabla retos_y_peliculas que proviene de la tabla peliculas.

El script que accede a la base de datos de retos_y_peliculas es:

- **conexion.php**: script que accede a la base de datos.
- **DAO_Reto.php**: clase que contiene funciones que realizan modificaciones y/o consultas en la tabla **retos**.

comentarios:

La finalidad de esta tabla es almacenar todo lo relacionado a la sección de comentarios. Aquí se detallan los campos que contiene:

-“**codigo**”: tipo entero para distinguir comentarios que auto incrementa según se vayan añadiendo.

-“**pelicula**”: clave externa en relación con “**pelicula**” de la tabla comentarios que proviene de la tabla peliculas.

-“**usuario**”: clave externa en relación con “**usuario**” de la tabla comentarios que proviene de la tabla users.

-“**valoracion**”: entero que sirve para detallar la puntuación dada por el usuario, entre 1 y 5.

-“**comentario**”: tipo text que contiene el comentario/crítica del usuario.

El script que accede a la base de datos de comentarios es:

- **conexion.php**: script que accede a la base de datos.
- **DAO_Comentarios.php**: clase que contiene funciones que realizan modificaciones y/o consultas en la tabla **comentarios**.

peliculas_actores:

La finalidad de esta tabla es almacenar los actores y su relación con las películas. Aquí se detallan sus campos:

-“**id_pelicula**”: clave externa en relación con “**id_pelicula**” de la tabla comentarios que proviene de la tabla peliculas.

-“**id_actor**”: clave externa en relación con “**id_actor**” de la tabla comentarios que proviene de la tabla actores_actrices.

El script que accede a la base de datos de películas_actores es:

- **conexion.php**: script que accede a la base de datos.

- **DAO_Pelicula.php**: clase que contiene funciones que realizan modificaciones y/o consultas en la tabla **peliculas_actores**.

actores_actrices:

La finalidad de esta tabla es almacenar l@s distint@s actores y actrices de las películas disponibles.

Aquí se detallan sus campos:

- “id”**: tipo entero que distingue los distintos actores entre sí.
- “nombre”**: tipo text que almacena el nombre del artista correspondiente.
- “edad”**: tipo entero que contiene la edad del artista correspondiente.

El script que accede a la base de datos de actores_actrices es:

- **conexion.php**: script que accede a la base de datos.
- **DAO_Pelicula.php**: clase que contiene funciones que realizan modificaciones y/o consultas en la tabla **peliculas_actores**.

INSTRUCCIONES DE INSTALACIÓN

Para utilizar la aplicación hay que seguir los siguientes pasos:

1. Descargar el paquete .zip que contiene el código fuente y la base de datos.
2. Activar el Xampp desde Xampp Control Panel: start "Apache" y start "MySQL".
3. Extraer el .zip en la carpeta htdocs.
4. Crear una base de datos vacía llamada "bd_watched"
5. Importar la Base de Datos llamada: "bd_watched.sql" de dentro del .zip
6. Acceder a la página escribiendo: localhost/Watched en la url de cualquiera navegador.
7. Para acceder a la administración de la Base de Datos: "localhost/phpmyadmin" en la url de cualquier navegador.
8. Disfrutar de la aplicación.

Para usarla en contenedor hay que seguir los siguientes pasos:

1. Abrir cualquier navegador.
2. Escribir: container.fdi.ucm.es:20101 en la url.
3. Repetir paso 8 de los anteriores pasos.

Es necesario tener un administrador creado en la Base de Datos para poder usar sus funcionalidades. No es posible registrarse como administrador. Solo como usuario normal.

Se pueden crear otros administradores mediante otro administrador.

Este tipo de usuario ya está creado con los siguientes datos:

Nombre: alex

Contraseña: alex

Aquí se enumeran los distintos usuarios de la Base de Datos rellena:

-sara
-pepe
-victor
-andrea
-juan
-laura
-pedro

Todos tienen la contraseña igual a su nombre. Por ejemplo: sara contraseña:sara

La tabla actores_actrices se debe rellenar a través de la herramienta de phpMyAdmin, debido a que el administrador no posee los derechos para modificar esta tabla.

Nuestra dirección del contenedor es: container.fdi.ucm.es:20101

Y nuestro usuario y contraseña para la Base de Datos en el servidor es:

Usuario: admin

Contraseña: iech7Urobith

A la cual se accede así: <http://container.fdi.ucm.es:20101/phpmyadmin/>

Hay dos bases de datos, una rellena con datos de prueba y otra vacía con un administrador. Ambas se tienen que usar con el mismo nombre: bd_watched
En la entrega, por tema de renombre, una se llamará bd_vacia, es solo cambiar el nombre para probarla.