

Informe de prácticas

Entrega 02

Alumno/a: Jordi Solé García

Alumno/a: Víctor Gallego Izquierdo

Contenido

Introducción.....	1
Decisiones de diseño.....	1
Package Application.....	1
Clase Image.....	1
Clase User.....	1
Package Driver.....	2
Driver Almacenamiento.....	2
Driver Conexión.....	2
Driver Imagen.....	2
insertarImagen.....	2
listImagesDB.....	3
searchImageID.....	3
modificarImagen.....	3
eliminarImagen.....	3
Driver User.....	4
checkUserPass.....	4
newUser.....	4
checkUser.....	4
Package Servlets.....	5
buscarImagen.....	5
eliminarImagen.....	5
login.....	6
modificarImagen.....	6
registrarImagen.....	6
registrarUsuario.....	6
Archivos web “.jsp”	7

buscarImagen.jsp.....	7
eliminarImagen.jsp.....	7
error.jsp.....	7
list.jsp.....	8
login.jsp.....	8
logout.jsp.....	8
menu.jsp.....	9
modificarImagen.jsp.....	9
mostrarImagen.jsp.....	9
register.jsp.....	10
registrarImagen.jsp.....	10
Repositorios de código consultados.....	11
Bibliografía consultada.....	11

Introducción

En este informe se van a detallar las diferentes funcionalidades del proyecto así como la implementación de los Servlets y archivos “.jsp” que corresponde a las páginas web.

Decisiones de diseño

Este proyecto ha sido realizado con una implementación con patrón MVC, con controladores para separar los Servlets del acceso a la base de datos y del guardado en disco de las imágenes. A continuación se va a explicar que contiene cada carpeta que se ha creado con los distintos archivos que usamos.

Package Application

Esta carpeta contiene dos archivos “.java” con las dos clases creadas para facilitar el manejo de los datos de cara a pasarlos como argumento.

Clase Image

Contiene un id, título, descripción, palabras clave, autor, creador, fecha de captura de la imagen, fecha de guardado de la imagen y el nombre del archivo. Tiene dos creadoras, una consiste en crear el objeto pasando todos los atributos y la otra sin añadirle ninguno.

Clase User

Contiene el nombre del usuario y la contraseña. Tiene dos creadoras, una consiste en crear el objeto pasando todos los atributos y la otra sin añadirle ninguno.

Package Driver

En esta carpeta se guardan todos los drivers que hemos implementado para el manejo de la base de datos y del guardado de las imágenes en disco. Al principio, pensamos en crear un único driver que manejara todo lo que tenga que ver con la base de datos pero al final decidimos separarlo en tres archivos distintos según tengan que ver con la conexión, las imágenes o los usuarios.

Driver Almacenamiento

Este driver consiste en una clase Java que sirve para controlar y gestionar el almacenamiento de las imágenes que se registran en la aplicación web. Contiene tres atributos fijo:

- String pathFiles : Fija la ruta absoluta del directorio donde queremos que se almacenen las imágenes.
- File uploads: Crea el atributo de tipo File a partir del atributo mencionado anteriormente.
- String extens[]: Consiste en un vector de strings que contiene las extensiones permitidas de las imágenes a registrar.

A continuación, contiene tres funciones:

- saveFile: que se ocupa de gestionar el almacenamiento del fichero.
- isExtension: controla las extensiones del fichero.
- DeleteFile: elimina el fichero del directorio.

Driver Conexión

Este driver se utiliza para crear la conexión con la base de datos. Tiene cuatro atributos que son constantes:

- URL → Dirección de la base de datos.
- dbUser → Usuario para acceder a la base de datos.
- dbPassword → Contraseña para acceder a la base de datos
- DRIVER → Driver que usa la aplicación para poder controlar la base de datos.

La única función que tiene es DBConexion, donde se le pasa un argumento de tipo conexión. En caso de no poder conectarse a la base, mostrará un error por consola indicándolo.

Driver Imagen

Este driver se usa para las consultas sobre las imágenes en la base de datos. Consiste en varias funciones.

insertarImagen

Dada una imagen, esta se introduce en la base de datos. Se prepara una introducción a la base de datos con los campos del objeto Imagen, excepto el id ya que este al crear la base de datos se actualiza automáticamente al introducirlo (incremental de uno en uno). En caso de error, se hace un catch de la excepción SQL y se muestra en consola. Al terminar, cierra la conexión con la base de datos automáticamente.

ListImagesDB

Función que retorna una lista de las imágenes que hay en la base de datos. Se prepara una query para seleccionar todos los datos que se encuentren en la tabla de imágenes para guardarlos en un ArrayList y retornarlos para ser mostrados. En caso de error, se hace un catch de la excepción SQL y se muestra en consola. Al terminar, cierra la conexión con la base de datos automáticamente.

searchImageID

Función que devuelve una imagen de la base de datos según el id introducido. Se prepara una query con una selección. En caso de error, se hace un catch de la excepción SQL y se muestra en consola. Al terminar, cierra la conexión con la base de datos automáticamente.

modificarImagen

Función que dada una imagen, actualiza los campos de esta en la base de datos. Se prepara una query con un update que permite actualizar los campos modificados. En caso de error, se hace un catch de la excepción SQL y se muestra en consola. Al terminar, cierra la conexión con la base de datos automáticamente.

eliminarImagen

Función que dado un id de una imagen, la elimina de la base de datos. Se prepara una query con un delete que elimina la imagen si el id corresponde con el introducido por el usuario. En caso de error, se hace un catch de la excepción SQL y se muestra en consola. Al terminar, cierra la conexión con la base de datos automáticamente.

BusquedaImagen

Función que busca una imagen ya registrada en la web a partir de, cómo mínimo, un dato de la imagen que no es generado automáticamente (título, descripción, palabras clave, autor y/o fecha de creación). Se prepara una query con los parámetros válidos de la función y se devuelve una lista de resultados. Si no hay resultados, esta es vacía. Finalmente, en caso de error se lleva a cabo un catch con excepción SQL y se muestra en consola. Al terminar, se hace el finally y se cierra conexión con la base de datos.

Driver User

Este driver se usa para las consultas sobre los usuarios en la base de datos.

CheckUserPass

Esta función comprueba, dado un nombre de usuario y su contraseña, si existe en la base de datos. En caso de existir y que la contraseña sea igual a la introducida se devuelve true. En caso de no existir o no coincidir las contraseñas se devuelve false. En caso de producir algún error durante la consulta, se hace un catch de este y lo muestra en consola.

NewUser

Esta función inserta en la base de datos un usuario con los parámetros del nombre y usuario del objeto. En caso de producir algún error durante la inserción, se hace un catch de este y lo muestra en consola.

CheckUser

Esta función comprueba si el usuario existe en la base de datos dado un nombre de usuario. Implementa una query con un Select para coger todos los usuarios e ir comprobando uno a uno si coincide con el id que se le pasa. En caso de existir devuelve false. En caso de producir algún error durante la consulta, se hace un catch de este y lo muestra en consola.

Package Servlets

Un servlet es una clase de lenguaje en Java que se usa en servidores. Nosotros al implementar una página web, los utilizamos para generarla de manera dinámica a partir de los parámetros de la petición que se envía a través del navegador desde donde accedemos. Hemos implementado un package donde se encuentran todos los servlets que usamos. En todos los servlets vamos a trabajar de una manera similar. Primero empezaremos comprobando que estemos dentro de nuestra sesión y que no hayamos accedido a la de nadie o simplemente sin iniciar sesión, excepto el login que no se comprueba la sesión. A continuación, haremos ya lo que se nos haya planteado con cada servlet y en caso de que ocurra algún error con el servlet, siempre se hace un catch de este error y se muestra siempre por consola.

buscarImagen

Servlet que recoge los datos que introduce el usuario y que corresponden a los datos(iguales o parecidos) de una imagen a buscar en la aplicación web.

Si el usuario intenta buscar una imagen sin introducir ningún campo en el formulario, la búsqueda se considera fallida y el servlet redirige a una página de error que mostrará el motivo del fallo. Además, dará la opción de volver a buscar o volver al menú. De lo contrario, se realiza la búsqueda con los campos introducidos y se imprime en la web lo siguiente:

- Si la búsqueda es satisfactoria: la lista de la imágenes resultantes, con la opción de modificar o eliminar si eres el propietario de estas.
- Contrariamente, se informa al usuario que la búsqueda no ha tenido resultados y se le da la opción de volver a buscar o de volver al menú.

Cabe añadir que hemos decidido imprimir los resultados de las búsqueda desde el servlet por comodidad.

En caso de fallo, se realiza un catch y se muestra el error por consola.

eliminarImagen

Servlet que recoge los datos de la imagen a eliminar y elimina tanto el fichero como la información de la base de datos. En caso de que la imagen no se pueda eliminar correctamente, se redirige a la página de error mostrando el error pertinente. En caso de que la imagen se elimine correctamente, se permitirá volver al menú.

El funcionamiento del servlet es el siguiente. El servlet obtiene de la pagina web el id de la imagen a eliminar y busca en la base de datos la imagen que corresponde a es id y lo obtiene. Se comprueba que esa imagen corresponda al mismo usuario que la ha introducido en la base de datos y que se pueda eliminar. En caso de que esto sea correcto, se elimina de disco y de la base de datos y se muestra un mensaje indicando que la imagen se ha borrado. En caso de que no se pueda eliminar de disco o que no sea el mismo usuario, se muestra un error.

login

Servlet que recoge los datos de la página de login y comprueba que el usuario esté en la base de datos. En caso de no estar en la base de datos, se redirige a la página de error mostrando el error pertinente.

El funcionamiento del servlet es el siguiente. Se obtiene el usuario y la contraseña introducidos en la página web, se comprueba que no sean nulos y se busca en la base de datos el usuario. En caso de ser alguno de los dos nulos, se redirige a la página de error y se muestra un error. Si el usuario existe se crea una sesión y se redirige al menú principal. En caso de no existir se redirige a la página de error indicando el error.

modificarImagen

Servlet que recoge los datos de la imagen a modificar y actualiza la base de datos y/o el fichero de la imagen.

El funcionamiento del servlet es el siguiente. Se obtienen todos los parámetros de la imagen que se puedan modificar y se comprueba que ninguno sea nulo, en caso de ser así se redirige a la página de error y se muestra un error. A continuación se comprueba si se ha subido un archivo de imagen, se guarda y se elimina el antiguo. En caso de fallo al eliminar el fichero antiguo se redirige a la página de error mostrando el error correspondiente. Finalmente se actualiza la tabla de base de datos de la imagen y se muestra opción de volver al menú principal.

registrarImagen

Servlet que recoge una imagen .jpg o .jpeg y sus datos relacionados introducidos por el usuario. Esta imagen la almacena en un directorio fijo de la aplicación web.

Por tanto, una vez controlada la sesión del usuario y se hayan recogido todos los datos el servlet gestiona tres posibles caso:

- Si falta algún campo para rellenar o no se ha subido la imagen, redirige a la página de error donde se muestra el motivo de este.
- Si los datos introducidos son correctos, se controla que la extensión de la imagen sea la correcta a partir del “DriverAlmacenamiento”. A continuación, se le fija el nombre que tendrá el fichero en sistema. Para esto, hemos decidido que el nombre contenga fecha, hora, minutos y segundos para conseguir que todas las imágenes sean únicas. Después guardamos en el directorio de la web a partir de la función “saveFile” de “DriverAlmacenamiento” y, finalmente, la insertamos en la base de datos con “insertarImagen” del “DriverImagen”. Además, se muestra por pantalla la opción de volver al menú o volver a buscar.
- Los datos introducidos están al completo, pero la extensión de la imagen no es la esperada. En este caso, también redirige a la página de error donde se muestra el motivo de este.

Finalmente, en caso de error se hace un catch de este y se muestra por consola.

registrarUsuario

Servlet que recoge los datos del usuario para registrarse en la base de datos. El usuario introduce un usuario y contraseña, escribiéndola dos veces para así confirmar que la ha escrito correctamente.

El funcionamiento del servlet es el siguiente. Se obtiene el usuario, la contraseña y la contraseña de confirmación y se comprueba que estos campos no sean vacíos. En caso de serlo se redirige a la página de error mostrando el error pertinente. Si estos no son vacíos, se comprueba que el usuario no exista y que ambos campos de la contraseña sean iguales. En caso de fallar algo, se redigire a página de error. Una vez se ha comprobado todo y es correcto, se registra al usuario y se le redirige automáticamente a la página de login.

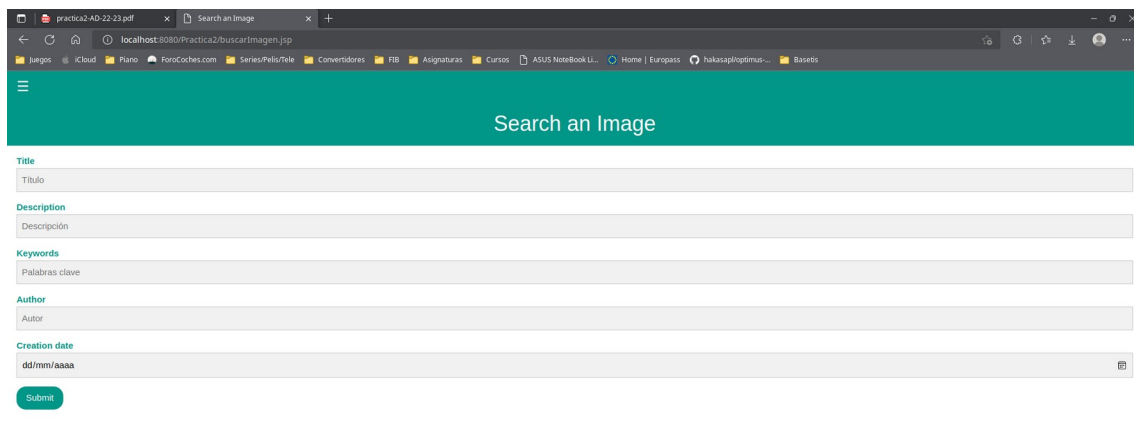
Archivos web “.jsp”

Estos archivos son los que el navegador web representa al usuario y con el que estos interactúan. Todos ellos, excepto el de error, están implementados con una alternativa a Bootstrap llamada **W3 CSS**. Esto está creado con el lenguaje CSS pero de manera que no tenemos que implementarlo nosotros sino que simplemente añadimos atributos a los botones, tablas... para que se representen según nos parezca mejor y según lo que dispongamos en la página web para usar.

De nuevo, todos los archivos “.jsp” implementan control de sesiones, excepto el login.jsp y register.jsp que no es necesario.

buscarImagen.jsp

Página que contiene el formulario para buscar una imagen a partir del título, la descripción, palabras clave, autor y/o fecha de creación. Esta página permite la conexión con su servlet correspondiente y este es el que se ocupa tanto de la gestión de resultados como de mostrarlos. Cabe destacar que a esta página solo se puede llegar a partir del menú o, habiendo hecho una búsqueda sin resultados dando opción a volver a buscar.



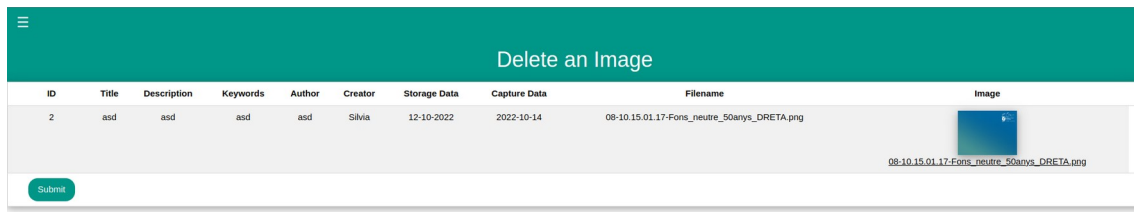
The screenshot shows a web browser window with the URL `localhost:8080/Practica2/buscarImagen.jsp`. The page has a teal header with the title "Search an Image". Below the header is a form with the following fields:

- Title**: Titulo
- Description**: Descripción
- Keywords**: Palabras clave
- Author**: Autor
- Creation date**: dd/mm/aaaa

At the bottom of the form is a green "Submit" button.

eliminarImagen.jsp

Página para eliminar una imagen del sistema. A esta página se puede llegar desde el listado de imágenes o desde la búsqueda de estas. En caso de querer ver la imagen más grande, se pulsa sobre ella y se abre una nueva página sobre la que se mostrará a mejor tamaño. Si se quiere eliminar, se pulsa sobre el botón de Submit y se eliminará. En la parte superior izq hay un botón que despliega un menú para poder dirigirse a otras páginas de la aplicación.

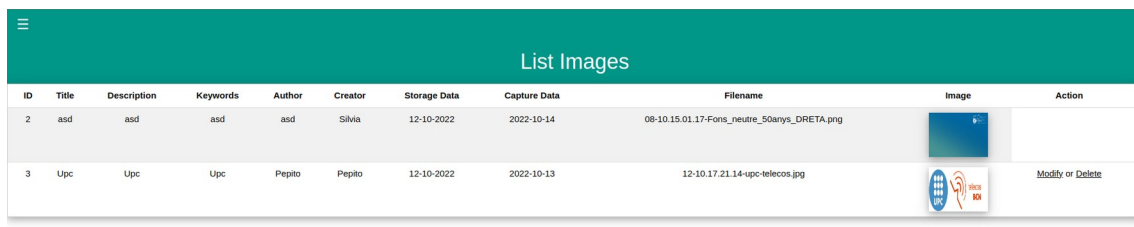


error.jsp

Página que indica que ha habido un error, mostrando cual, y muestra un enlace para volver a la página de login o menú según el tipo de error que se haya cometido y de la página de la que veníamos.

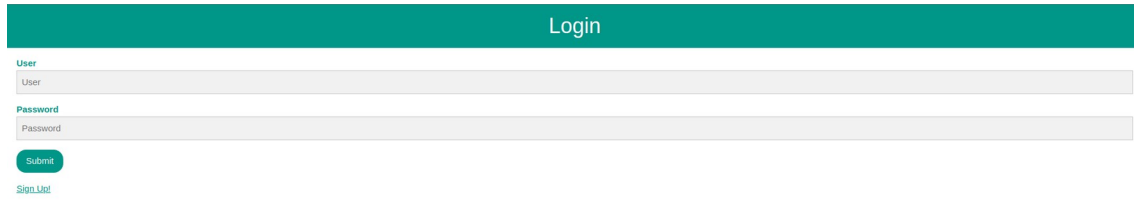
list.jsp

Página que lista todas las imágenes de la base de datos. En caso de querer ver la imagen más grande, se pulsa sobre ella y se abre una nueva página sobre la que se mostrará a mejor tamaño. Si se quiere eliminar o modificar y se es el usuario que la ha introducido en la base, se muestra la opción de modificar o eliminar. En la parte superior izq hay un botón que despliega un menú para poder dirigirse a otras páginas de la aplicación.



login.jsp

Página que contiene el formulario que pide el nombre de usuario y contraseña para acceder al sistema.



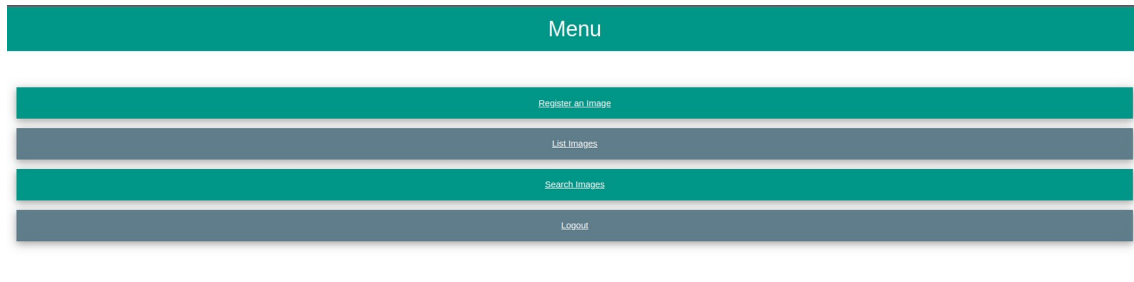
The screenshot shows a web form titled "Login" with a teal header. Below the header, there are two input fields: "User" and "Password". The "User" field has a placeholder text "User". The "Password" field has a placeholder text "Password". Below the "Password" field, there is a green "Submit" button and a link labeled "Sign Up!".

logout.jsp

Página que se accede desde cualquier otra, excepto el login, que realiza el logout de la sesión invalidándola y redirigiendo de nuevo al login sin llegar a ver nada ni mostrar nada por pantalla.

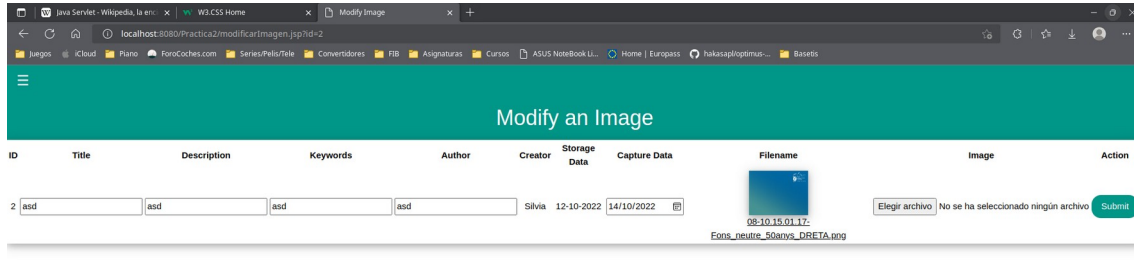
menu.jsp

Página que muestra el menú de opciones del usuario una vez ha iniciado sesión. Solo contiene enlaces a registrar una imagen, listar las imágenes, buscar imágenes y hacer el logout.



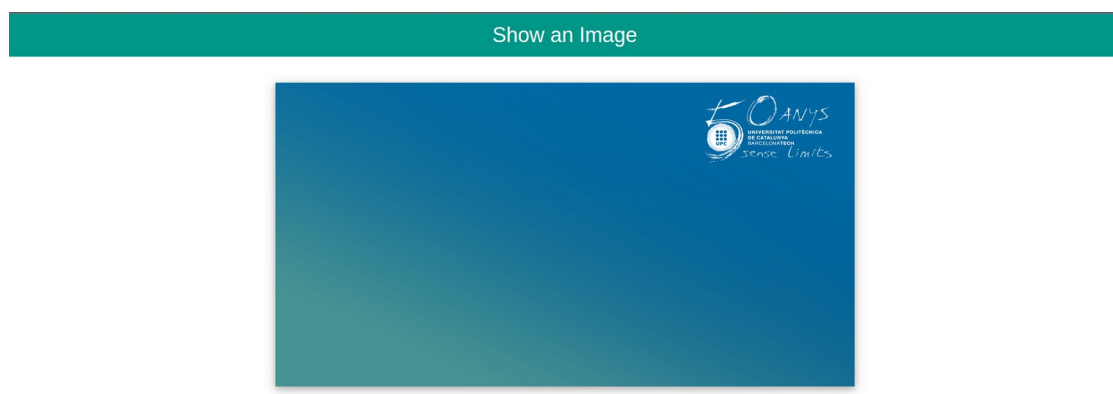
modificarImagen.jsp

Página que permite modificar los datos de una imagen registrada por el usuario. A esta página se puede llegar desde la página de listar las imágenes o desde la página de buscar imágenes. Para aceptar el cambio de los datos se tiene que presionar el botón de Submit. En la parte superior izq hay un botón que despliega un menú para poder dirigirse a otras páginas de la aplicación.



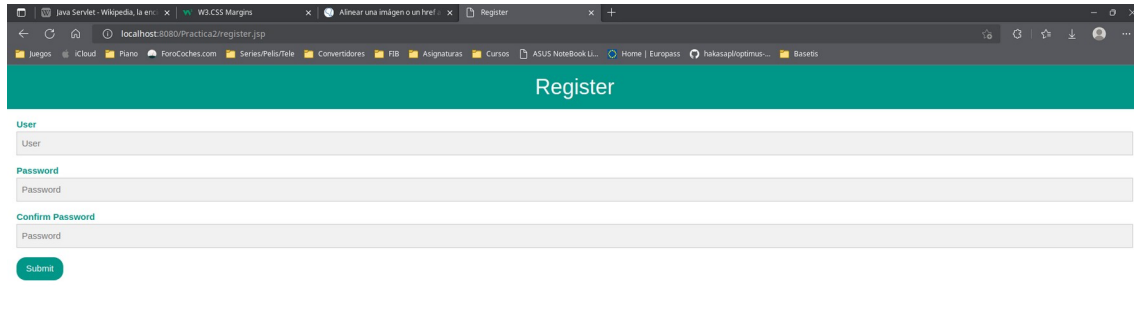
mostrarImagen.jsp

Página que permite mostrar la imagen a tamaño mayor al usuario. A esta página se accede desde el listado de imágenes, modificar imagen, eliminar imagen y desde la búsqueda de imágenes.



register.jsp

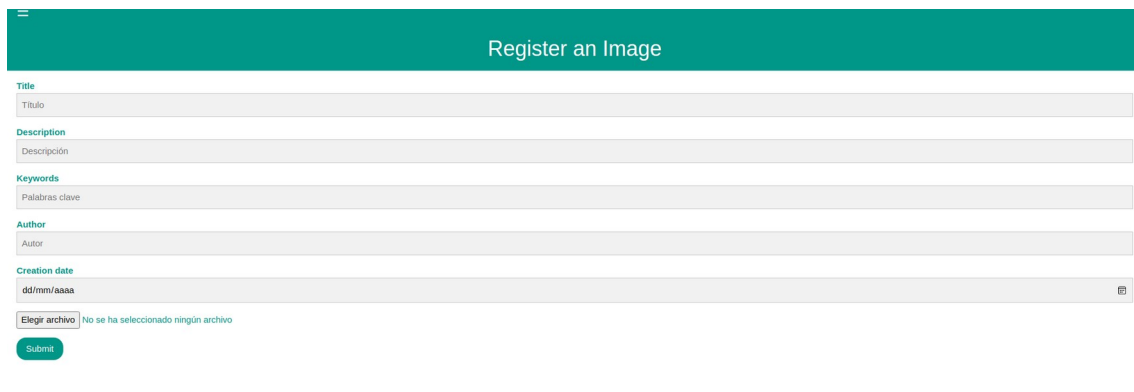
Página que te permite registrarte en la base de datos para poder acceder al sistema. A esta página solo se puede acceder desde el login.



The screenshot shows a web browser window with the URL `localhost:8080/Practica2/register.jsp`. The page has a teal header with the title "Register". Below the header, there are three input fields: "User", "Password", and "Confirm Password". Each field has a small label above it. At the bottom of the form, there is a green "Submit" button.

registrarImagen.jsp

Página que pide los datos de una imagen para darla de alta en la aplicación. Para introducir la fecha de captura, se puede tanto escribir directamente como pulsar sobre el calendario y seleccionarla. En caso de querer registrar la imagen en la base de datos, es necesario pulsar el botón de Submit una vez rellenados todos los campos. En la parte superior izq hay un botón que despliega un menú para poder dirigirse a otras páginas de la aplicación.



The screenshot shows a web browser window with the URL `localhost:8080/Practica2/registrarImagen.jsp`. The page has a teal header with the title "Register an Image". Below the header, there are several input fields: "Title", "Description", "Keywords", "Author", and "Creation date". Each field has a small label above it. At the bottom of the form, there is a green "Submit" button. There is also a file upload section with a button labeled "Elegir archivo" and a message "No se ha seleccionado ningún archivo".

Repositorios de código consultados

Lista de ejemplos de código consultados en github, gitlab, etc.

Bibliografía consultada

Lista de webs consultadas para la realización de la práctica que no sean repositorios de código.

W3.CSS Home. (s. f.). Recuperado 12 de octubre de 2022, de <https://www.w3schools.com/w3css/default.asp>

León, D. P. de. (s. f.). Tablas en HTML. Recuperado 12 de octubre de 2022, de <https://www.htmlquick.com/es/tutorials/tables.html>

pildorasinformaticas. (s. f.). YouTube. Recuperado 12 de octubre de 2022, de <https://www.youtube.com/c/pildorasinformaticas>

Stack Overflow. (s. f.). Stack Overflow. Recuperado 12 de octubre de 2022, de <https://es.stackoverflow.com/>