Aplicaciones Distribuidas (AD)

Práctica 2: Desarrollo de una aplicación web

En esta práctica se va a desarrollar una aplicación web con el IDE *Apache Netbeans*. La práctica durará varias sesiones y deberá realizarse una única entrega. Hay que entregar el código desarrollado junto con un informe del trabajo realizado.

En este informe se tiene que explicar el funcionamiento de la práctica, incluyendo funcionalidades extra añadidas y posibles modificaciones a la estructura de la aplicación presentada en este enunciado. Se tiene que entregar todo en un único archivo comprimido.

Revisad la rúbrica de evaluación para saber qué aspectos se van a evaluar en esta práctica y su peso sobre la nota de la misma. Implementad los máximos puntos de la rúbrica porque las funcionalidades que se piden se utilizarán en las siguientes prácticas.

Consultad la fecha de entrega en el Racó.

Desarrollo de la práctica

La práctica consiste en desarrollar una aplicación web para gestionar imágenes (pueden ser de distintos tipos, JPEG, GIF, PNG, etc.) por parte de los usuarios registrados. Con esta aplicación los usuarios podrán hacer login, registrar nuevas imágenes, modificar o eliminar los registros de las imágenes que hayan creado y buscar imágenes a partir de sus características. La información se debe almacenar en una base de datos como la de la práctica 1. Podéis añadir los campos que consideréis necesarios a las tablas de la base de datos.

Es obligatorio implementar sesiones (`HttpSession`[1]) para poder acceder a las funcionalidades del sistema. En caso de que un usuario intente acceder sin haber hecho `login`, será redirigido a la página inicial, `login.jsp`. Es necesario que todas las páginas sean de tipo `jsp` para poder comprobar si el usuario tiene una sesión abierta o no.

Se deben implementar (como mínimo) los siguientes componentes (páginas jsp y servlets):

- ***login.jsp***: Página que contiene el formulario que pide los datos al usuario para entrar en el sistema. Estos datos son usuario y *password*.
- ***login.java***: Servlet que recoge los datos de la página de `login`, comprueba que el usuario esté en la base de datos. Si está, nos redirecciona al menú. Si no está o el *password* es erróneo debe informar al usuario con la página de error y darle la opción de ir al `login`.

[1] https://jakarta.ee/specifications/platform/9/apidocs/jakarta/servlet/http/httpsession

- ***menu.jsp***: Página que muestra el menú de opciones cuando el usuario introduce correctamente sus datos. Esta página sólo contiene enlaces a las páginas jsp de gestión de imágenes que se detallan a continuación.
- ***registrarImagen.jsp:*** Página que pide los datos de una imagen para darla de alta en la aplicación. Los datos de una imagen deben ser (como mínimo), identificador de la imagen (no se le pide al usuario, se asigna automáticamente), título, descripción, palabras clave, autor (el usuario que capturó la imagen inicialmente), creador (el usuario que la inserta en el sistema), fecha de creación, fecha de alta en el sistema (esta no se pide al usuario) y nombre del fichero que la contiene. Los datos de la imagen que no se generen automáticamente se tienen que pedir al usuario con un formulario. El fichero de imagen también se tiene que subir al sistema (campo de formulario HTML tipo file) y se almacenará en un directorio fijo de la aplicación web (ver Anexo 1 para detalles de gestión de los directorios). Los campos autor y creador pueden ser el mismo, aunque sólo el creador tiene que ser un usuario dado de alta en la base de datos (*foreign key* en las tablas de la práctica 1).
- ***registrarImagen.java***: Servlet que recoge los datos de la imagen y los guarda en la base de datos. También debe leer la imagen y guardarla en un directorio de la aplicación web. En caso de que se pueda registrar la imagen correctamente, mostrará un mensaje y dará la opción de volver al menú o registrar otra imagen. En caso de error, redireccionará al usuario a la página de error.
- ***modificarImagen.jsp:*** Página que permite modificar los datos de una imagen registrada por un usuario en el sistema. A esta página se puede llegar desde la página de búsqueda, que se explica más adelante.
- ***modificarImagen.java***: Servlet que recoge los datos de la imagen a modificar y actualiza la base de datos y/o el fichero de la imagen. En caso de que se pueda modificar la imagen correctamente, mostrará un mensaje y dará la opción de volver al menú. En caso de error, redireccionará al usuario a la página de error.
- ***eliminarImagen.jsp:*** Página para eliminar una imagen del sistema. A esta página se puede llegar desde la página de búsqueda, que se explica más adelante. El fichero asociado a los datos que hay en la base de datos también se tiene que eliminar.
- ***eliminarImagen.java***: 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 se pueda eliminar la imagen correctamente, mostrará un mensaje y dará la opción de volver al menú. En caso de error, redireccionará al usuario a la página de error.
- ***buscarImagen.jsp:*** Página que permite buscar imágenes a partir de uno o varios campos asociados a la imagen. Si no se utiliza ningún campo de búsqueda, tienen que aparecer todas las imágenes. Para las imágenes que sean del usuario que hace la búsqueda tiene que aparecer la posibilidad de modificar la imagen (enlace a la página modificarImagen) o eliminarla (enlace a la página eliminarImagen).
- ***buscarImagen.java***: Servlet que recoge los datos de búsqueda y devuelve el resultado de la búsqueda en forma de listado. Si la búsqueda no devuelve ningún resultado se debe informar al usuario, dándole la posibilidad de realizar otra búsqueda o de volver al menú.
- ***error.jsp/java***: Página que indica que ha habido un error y muestra un enlace para volver a la página de login o al menú, dependiendo de la página de la que vengamos.

El flujo básico entre páginas se muestra en la Figura 1 (podéis añadir más páginas y/o conexiones entre páginas si lo creéis necesario):

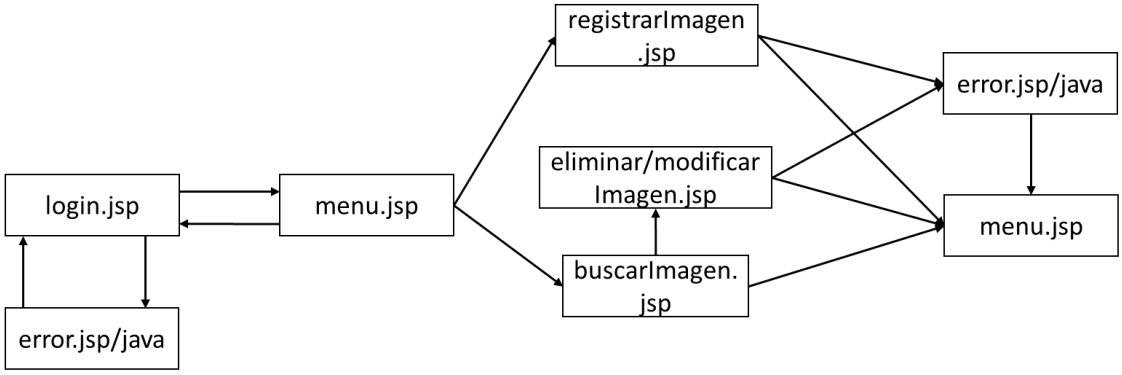

Figura 1. Flujo entre páginas

Algunas consideraciones:

- En la Figura 1 no aparecen los servlets que recogen los datos de los formularios que hay en las páginas jsp. Por ejemplo, para ir de `login.jsp` a `menu.jsp` es necesario pasar por un servlet (`login.java`) que compruebe que el usuario existe en la base de datos.
- Se pueden añadir más funcionalidades a la aplicación como por ejemplo el registro de usuarios o el logout.
- Se puede cambiar el servlet `error.java` por una jsp.
- Se pueden incluir hojas de estilo para modificar el aspecto de las páginas. El uso de hojas de estilo (CSS) tendrá un impacto pequeño en la nota de la práctica. Si la práctica no funciona, no se aprobará esta entrega por usar hojas de estilo.
- Todas las modificaciones realizadas se deben explicar en el informe de la práctica.

Algunas referencias relacionadas con subida y gestión de ficheros en Aplicaciones Web J2EE

- https://docs.oracle.com/javaee/6/tutorial/doc/glraq.html
- https://www.erpgreat.com/java/upload-files-using-jsp-and-servlet.htm
- https://docs.oracle.com/javaee/7/tutorial/servlets016.htm
- https://stackoverflow.com/questions/8885201/uploaded-image-only-available-after-refreshing-the-page (para resolver el problema de tener que recargar la aplicación después de subir las imágenes)
- https://stackoverflow.com/questions/18664579/recommended-way-to-save-uploaded-files-in-a-servlet-application (para que todas las imágenes se almacenen en un directorio común del sistema)