PROGRAMACIÓN EN INTERNET

ENTREGA 03 :

- **API REST**
- **ANGULAR JS**

NOMBRE: JUAN LUIS FRAGOSO DEL REY

DNI : 52969926-Y

TITULACIÓN : GRADO EN INGENIERÍA INFORMÁTICA DEL SOFTWARE.

ÍNDICE:

INTRODUCCIÓN

En esta entrega se lleva a cabo la programación de nuestra aplicación del lado del cliente. Para ello implementaremos una API REST que nos permitirá interaccionar con nuestra base de datos sin necesidad de usar servlets. Por otro lado, para realizar la presentación de las vistas y el manejo de los datos devueltos por los servicios REST, utilizaremos el framework AngularJS, de manera que nuestra aplicación permanezca estática como si de una aplicación de escritorio se tratara, pero interaccionando con el cliente de forma dinámica.

REST

Los servicios ofrecidos mediante REST se implementan basándonos en Jersey, que es una API de Java que permite la creación de servicios web de este tipo. Al solicitar un servicio, nos devolverá un resultado en formato JSON. Podemos dividir la API REST en 3 categorías o secciones:

Noticias

Mediante peticiones GET, se permite obtener todos las noticias registradas en la base de datos, una noticia a partir de su identificador, las noticias pertenecientes a un usuario concreto y noticias filtradas, es decir, podemos obtener noticias pertenecientes a una categoría concreta, las noticias más visitadas, las más populares o las más destacadas (más comentadas).

Para acceder a los recursos de las noticias disponemos de las siguientes urls:

- Todas las noticias:
  - http://localhost:8080/meneame/rest/news
- Acceder a una noticia concreta perteneciente al usuario logueado:
  - http://localhost:8080/meneame/rest/news/*id_noticia*
- Acceder a una noticia concreta por su id independiente del usuario logueado:
  - http://localhost:8080/meneame/rest/detalle/news/*id_noticia*
- Obtener noticias por categoría:
  - http://localhost:8080/meneame/rest/news/*categoria*
- Obtener noticias más visitadas:
  - http://localhost:8080/meneame/rest/news/visitadas
- Obtener noticias más populares:
  - http://localhost:8080/meneame/rest/news/populares
- Obtener noticias más destacadas o comentadas:
  - http://localhost:8080/meneame/rest/news/destacadas

Las funciones de obtener noticia por id están diferencias en función de su el usuario está logueado o no, ya que, de no ser así, se produciría un fallo de seguridad que permitiría a un usuario logueado acceder a una noticia que no es suya y modificarla.

Mediante una petición POST, se puede realizar la inserción de una nueva noticia, su url es http://localhost:8080/meneame/rest/news.

Al realizar una petición PUT, se puede modificar una noticia ya existente, pasando su id como parámetro por la url, que quedaría definida como http://localhost:8080/meneame/rest/news/*id_noticia* .

Igual que se modifica una noticia con una petición PUT, podremos borrarla a través de una petición DELETE, con la url estructurada de la misma manera que en la petición PUT.

Usuarios

Mediante peticiones GET se puede obtener todos los usuarios registrados en la base de datos, un usuario determinado a través de su id o a través de su nombre y el usuario de la sesión. Estas solicitudes no devuelven las contraseña de los usuarios, ya que cualquier persona usuario podría acceder a ellas y ver la contraseña de otros usuarios.

Para acceder a los recursos de los usuarios mediante peticiones GET tenemos las siguientes urls:

- Obtener todos los usuarios de la base de datos:
  - http://localhost:8080/meneame/rest/users
- Obtener un usuario mediante su id:
  - http://localhost:8080/meneame/rest/users/*id_usuario*
- Obtener un usuario mediante su nombre:
  - http://localhost:8080/meneame/rest/users/*nombre_usuario*
- Obtener el usuario de la sesión:
  - http://localhost:8080/meneame/rest/users/getUser

Al realizar una petición POST, se puede insertar un nuevo usuario en la base de datos que recibe a través de JSON. Su url es http://localhost:8080/meneame/rest/users .

Al realizar una petición PUT o DELETE, podemos modificar o borrar, respectivamente, un usuario a través de su id que recibe la url como parámetro. Por tanto, la url quedará formada como http://localhost:8080/meneame/rest/users/*id_Usuario* .

Comentarios

Mediante peticiones GET se puede obtener todos los comentarios registrados en la base de datos, un comentario determinado a través de su id y todos los comentarios pertenecientes a un noticia o un usuario determinado.

Para acceder a los recursos de los comentarios a través de peticiones GET tenemos las siguientes url:

- Obtener todos los comentarios de la base de datos:
  - http://localhost:8080/meneame/rest/comments
- Obtener todos los comentarios pertenecientes a una noticia determinada:
  - http://locahost:8080/meneame/rest/comments/news/*id_noticia*
- Obtener todos los comentarios escritos por un usuario determinado:
  - http://locahost:8080/meneame/rest/comments/owner/*id_usuario*

- Obtener un comentario a través de su id:
    - http://localhost:8080/meneame/rest/comments/*id_comentario*

Al realizar una petición POST sobre la url http://locahost:8080/meneame/rest/comments se puede insertar un comentario nuevo en la base de datos. Los datos pertenecientes al nuevo comentario se reciben en formato JSON.

Al realizar una petición PUT o DELETE sobre la dirección http://locahost:8080/meneame/rest/comments/*id_comentario* podremos modificar o borrar el comentario cuyo id pertenece al pasado por parámetro respectivamente.

SEGURIDAD

El proceso de login y acceso a contenidos privados se continuúa realizando mediante servlets, ya que no es requisito para esta entrega que sea implementado en AngularJS y REST. En cambio, es necesario destacar los cambios que se han hecho en el servlet "*LoginServlet"* y *"LogOutServlet"* para adaptarlo a las nuevas funcionalidades.

En primer lugar,en el filtro de la aplicación que controla el acceso a contenidos privados, implementamos el método doFilter() de la siguiente manera:

```
public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain)
        throws IOException, ServletException {
    // TODO Auto-generated method stub
    HttpServletRequest req = (HttpServletRequest) request;
    HttpServletResponse res = (HttpServletResponse) response;
    HttpSession session = ((HttpServletRequest) request).getSession(true);

    StringBuffer url = new StringBuffer("http://localhost:8080/meneame/rest/users/");

    if (session.getAttribute("user") != null || req.getMethod().equalsIgnoreCase("GET")
            || (req.getMethod().equalsIgnoreCase("POST")
                    && url.toString().equals(req.getRequestURL().toString()))) {
        chain.doFilter(request, response);
    } else {
        res.sendRedirect("/LoginServlet");
    }
}
```

En este método se realiza las comprobaciones necesarias para permitir el acceso a los recursos en los siguientes casos:

- Si se realiza una petición GET, ya sea sobre noticias, usuarios o comentarios.
- Si se realiza una petición POST sobre la url http://locahost:8080/meneame/rest/users , es decir, si se intenta registrar un nuevo usuario.

El resto de las peticiones tienen acceso restringido, y únicamente podrán realizarse en caso de que un usuario esté registrado. Si el usuario no está registrado e intenta acceder a alguno de estos recursos, será redirigido a la página de login.

ANGULAR JS

El uso de AngularJS permite que, a través de los html de la entrega 01, estos sean enriquecidos y gestionados de tal manera que la página siga el modelo Single Page Application, es decir, la aplicación web nunca abandona la página principal.

Para realizar esto, se define una página index.html, donde se define la cabecera y el footer de nuestra web y, gracias a la etiqueta ng-view, cargará un contenido u otro en función de la url que este activa en ese momento.

Factorías

Las factorías enlazan con los recursos ofrecidos a través de REST, y gracias a ellas, las funciones implementadas en los controladores pueden acceder a ellos.
Tenemos 3 factorías que acceden a los recursos de cada sección de REST, es decir:

- Factoría *newsFactory* que accede a los recursos de noticias.
- Factoría *userFactory* que accede a los recursos de usuarios.
- Factoría *commentFactory* que accede a los recursos de comentarios.

Controladores

Nuestra aplicación contiene los siguientes controladores:

- mainAppCtrl → Se encarga de gestionar la información que se muestra en la cabecera, es decir, si un usuario está logueado, muestra su nombre.
- listNews → Se encarga de listar todas las noticias disponibles en la base de datos, ya sea en la página principal, o en las distintas categorizaciones realizadas.
- newsCtrl → Gestiona las operaciones CRUD de las noticias, es decir, gestiona la inserción, modificación y borrado.
- userCtrl → Gestiona las operaciones CRUD de los usuarios, es decir, gestiona la inserción, modificación y borrado.
- commentsCtrl → Gestiona las operaciones CRUD de los comentarios, es decir, gestiona la inserción, modificación y borrado.

Reutilización de archivos html

Para listar las noticias, ya sea por categoría o el listado de noticias general, se utiliza siempre el mismo archivo html, llamado listNews.html

Para realizar una inserción de una noticia o editarla, se utiliza siempre el mismo archivo html llamado enviarNoticia.html.

Para realizar la inserción de un usuario o modificarlo, se utiliza el fichero nuevoUsuario.html.

A la hora de insertar un nuevo comentario o modificarlo, se realiza a través del html llamado nuevoComentario.html.

OPCIONES EXTRAS

Se ofrecen distintas funcionalidades extras, enumeradas a continuación:

- Se incluye un buscador de noticias que permite filtrar las mismas en función del texto escrito en el cuadro de búsqueda.
- Se incluye listado de noticias por categorías, que pueden ser:
    - Ciencia y Tecnología.
    - Deportes.
    - Noticias más visitadas.
    - Noticias más populares (más votadas).
    - Noticias destacadas (con mayor número de comentarios).

Para la entrega final se realizará la implementación correspondiente para permitir votar noticias y registrar las visitas a cada noticia, ya que debido a la falta de tiempo no se ha podido realizar en esta entrega. Por el mismo motivo, no se ha implementado la funcionalidad extra de que cada usuario pueda acceder a una sección que englobe todos los comentarios escritos por el en cualquier noticia. Esta funcionalidad también se implementará en la entrega final.

Por último, a pesar de que no es necesario, en la entrega final también se intentará realizar el control de seguridad y login a través de AngularJS y no desde servlets.

PRUEBAS

Se puede comprobar el correcto funcionamiento de la aplicación web a través del vídeo que se encuentra en la siguiente url:

https://www.youtube.com/watch?v=wBJHY17b1JM&feature=youtu.be

CONCLUSIONES

El uso de AngularJS me ha parecido bastante interesante, ya que permite realizar funcionalidades que con las herramientas utilizadas en las entregas anteriores eran más difíciles como, por ejemplo, el buscador de noticias. El uso de scope para pasar valores entre vista y controlador me ha parecido bastante útil, aunque a mi parecer podría haberlo explotado más en mi práctica e intentaré hacer más uso de ello en la entrega final.