# Pontificia Universidad Católica Madre y Maestra Facultad de Ciencias de la Ingeniería

**Departamento de Ingeniería en Sistemas y Computación**



**Programación Web ST•ISC•415•T•001**

#### Practica #3:

Creación de Blog

#### Estudiante:

Daniel Pérez, 2011-0839

Ariel Salce, 2009-0707

#### Profesor:

Carlos A. Camacho G.

#### Fecha de entrega:

Viernes, 3 de Junio del 2016

Santiago de los Caballeros, República Dominicana

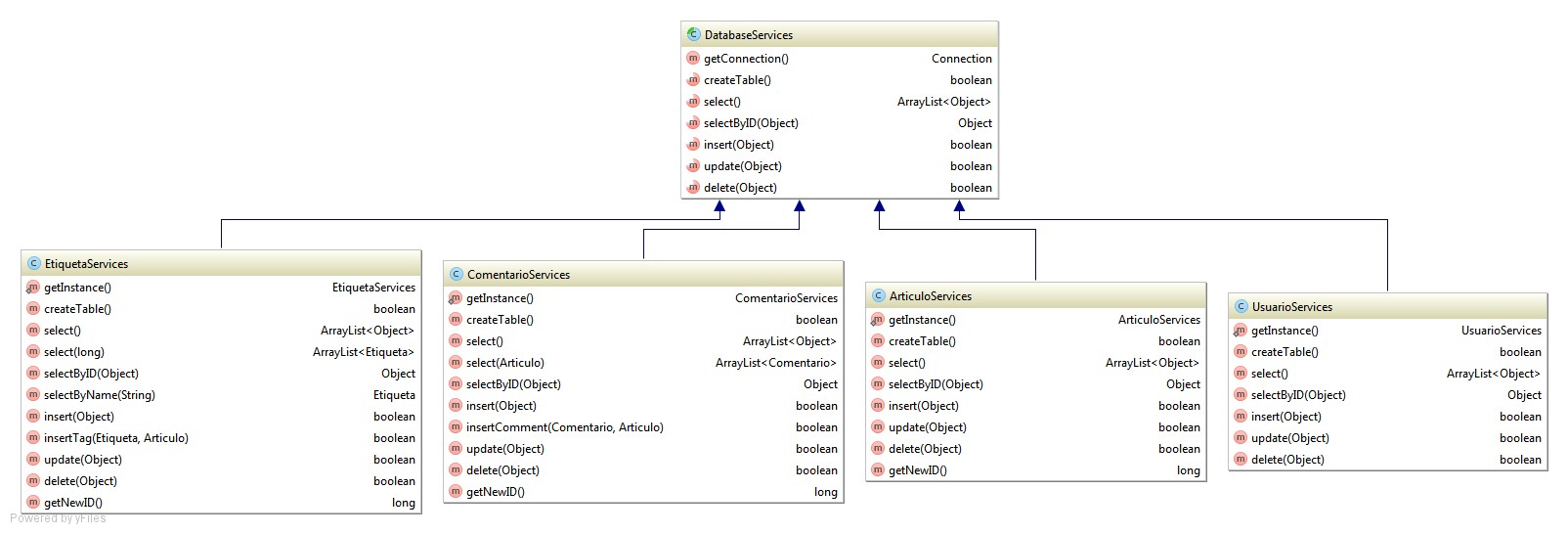
Introducción

Para la práctica de esta semana, se pidió por el profesor la realización de un blog utilizando el micro- framework SparkJava (utilizado en la práctica anterior), el framework de front-end Bootstrap 3 y el sistema de manejo de base de datos relacionales H2, en el que se puedan visualizar distintos artículos que serán posteados en el blog. Se tendrán distintos usuarios que se pueden crear como lectores o autores, solo los autores pueden crear nuevos artículos. También se podría comentar en los artículos ambos tipos de usuarios pueden comentar en un artículo. Existe un tipo de usuario administrador que puede borrar artículos no importa quién lo haya creado mientras que un autor puede borrar solo sus propios artículos. Para todo lo relacionado a permisos especiales se van a utilizar filtros para la navegación alrededor de la aplicación Web y con respecto a la persistencia (Login de los usuarios, por ejemplo) el concepto de Sesión y Cookies.

## Desarrollo

### Servicios de Base de Datos

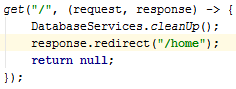
Para empezar, se decidió crear diferentes servicios de base de datos para cada entidad por separado para que el código Java sea más legible y sea más fácil de implementar, especialmente porque la práctica no solo fue realizada por una persona y para que otra persona entienda como usar los servicios escritos por alguien más era necesario que fueran lo más fácil posible de utilizar. Este es el diagrama UML de los servicios de base de datos utilizados para este programa:



Luego de tener los servicios, se procedió crear la lógica del sitio web. Todo con respecto a la lógica se realizó utilizando SparkJava, con las funciones GET y POST, en combinación con FreeMarker.

### Inicio de la aplicación

Para este punto, se utilizó el URL por defecto que fuese “/”, donde al ir a este, se borran comentarios, etiquetas, artículos, usuarios y demás componentes usados en la aplicación, se crea un usuario administrador por defecto con el nombre, el username, el apellido como “USER”, la contraseña como “ADMIN” y las características de si es administrador y autor ambas como verdaderas, de tal forma que este usuario tenga total dominio de todas las características en la aplicación. Luego de todo esto, nos manda automáticamente a la página principal de la aplicación.

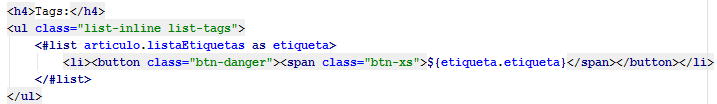


El método ***CleanUp*** es el que accede a la base de datos para borrar las entidades antes mencionadas y agregar al usuario administrador por defecto al sistema.

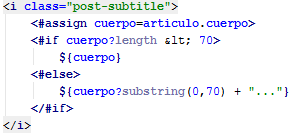
### Página Principal de la Aplicación

En esta página se manejan ciertas pautas expuestas por el profesor, entre estas están:

* **Mostrar todos los artículos, donde los primeros son los nuevos y van hasta los más viejos:** Lo primero que se debe hacer obtener todos los artículos de la base de datos. Después, se ordenan utilizando ***Collections.sort*** utilizando el método de la clase *Comparable* llamado “***compareTo***” para comparar los artículos por fecha, y se revierten ya que ***Collections.sort*** los ordena de la manera contraria a la que se necesita.
* **Se deben visualizar las etiquetas de cada artículo en la página principal:** Todo lo demás se maneja en FreeMarker dándole lo necesario al Template de la página principal con un Diccionario (HashMap). Las Etiquetas se encuentran en una lista dentro de cada Artículo, por lo que se puede acceder a ellas para mostrarla cuando se tiene cada Artículo. Por esto y otras razones, todos los Artículos se le mandan al Template de la página principal.



* **El texto a presentar en el resumen de la página principal de cada artículo, debe no exceder los 70 caracteres y se debe incluir un enlace al artículo completo:** Esta es otra de las razones por la que se mandan al Template principal los Artículos, ya que de aquí se puede extraer el cuerpo de cada uno de los Artículos y utilizando código de FreeMarker en este caso, se puede limitar a que solo se muestren hasta 70 caracteres de cada Artículo en la página.



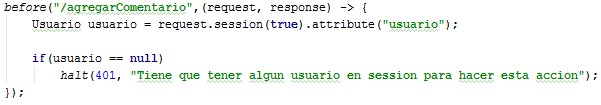
Para lograr tener un enlace que redirija al post completo, solo se colocó un botón, el cual manda al método GET que va a mostrar el Articulo completo el id del artículo, para que este se encargue de obtener lo necesario de los artículos para mostrar todo su contenido.



### Página del Artículo

* **En la página del Artículo se deben mostrar todos los comentarios de este, su contenido, un formulario para postear comentarios donde solo los usuarios autentificados puedan agregar comentarios:** El primer paso para realizar lo pedido es, en el método GET de la página del Artículo completo obtener la lista de Comentarios y Artículos de la base de datos, para mostrar su contenido utilizando el Template de la página del Artículo y ambas listas se obtienen con el ID del artículo obtenido desde la página principal.

La parte de donde solo los usuarios autenticados agregan comentarios se realiza utilizando lo que son los filtros. Estos solamente necesitan la url de destino a “filtrar”, y se manejan por una expresión Lambda con request y response. Se puede ver que tienen una sintaxis parecida con los métodos GET y POST que se han usado en las practicas anteriores y esta, lo único que en lugar de GET o POST, se le coloca ***before*** o ***after***, para denotar si va a filtrarse antes o después de realizar la acción determinada. Dentro de estas funciones, solo se agrega la lógica que se necesita para el filtro especificado, por ejemplo, en este caso se colocó lo siguiente:

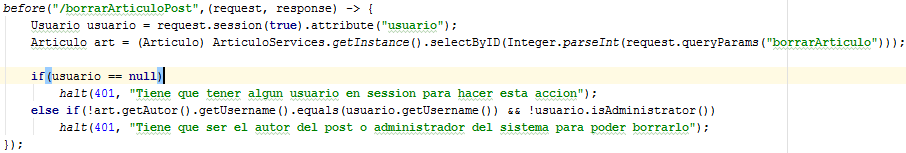


Este URL utilizado es el que tiene el botón que permite el postear comentarios en la página del artículo, por lo que cuando se dé clic en el botón, se verifica que algún usuario existe en la sesión actual, es decir si algún usuario está logeado y de no ser así se muestra un mensaje de error que informe a los que utilizan la aplicación que deben estar logeados en el sistema para postear comentarios. De esta forma se realizan los filtros para las demás funcionalidades del sistema que los requieran. Para los filtros también cabe mencionar que se creó una clase específica para darle mejor claridad y orden a las cosas. En esta clase se encuentran todos y cada uno de los filtros utilizados como el de *agregarComentario* que se mostró anteriormente.



### Página del Administrador

* **El usuario administrador y los autores pueden borrar artículos que consideren no apropiados. Además, el usuario administrador y los autores pueden crear o modificar un artículo, donde deben indicar el título, cuerpo y las etiquetas para este:** Para el usuario administrador, se creó una clase de nombre ZonaAdmin, donde se encuentra todo lo relacionado a las acciones específicas que este usuario con privilegios especiales puede realizar y sus interfaces Web. Volviendo al tema de los filtros, el que permite que solo autores o el administrador borre algún Artículo es el siguiente:



Como podemos ver nuevamente, la misma sintaxis mostrada anteriormente, pero condiciones distintas en el cuerpo para manejar lo exigido. Lo mismo se hace para filtrar que solo los usuarios autores o administradores sean los únicos que pueden crear algún artículo o modificarlo. En la creación de los artículos simplemente se crean 3 tags HTML, que son: 1 *input* para el título, un *textarea* para el cuerpo y otro *input* para las etiquetas.

### Repositorio (GitHub)

<https://github.com/danieldmw2/ProgramacionWeb1>

## Conclusión

El concepto de filtros fue nuevo para nosotros y como era desconocido al principio nos sentimos un poco incomodos con este, pero luego nos dimos cuenta lo sencillo y útiles que son y pudimos incorporarlos relativamente rápido al proyecto, de una manera limpia y concisa. También, fue interesante el poder aplicar muy fácilmente los conceptos de manejo de sesiones y cookies al sistema.

En general, esta práctica tuvo ciertas dificultades con respecto al tiempo que tenía como límite para ser terminada que fue relativamente corto para tantas funcionalidades, por lo que se nos apretaron bastante las demás materias durante la semana. Una de las cosas que nos pudieron ahorrar un poco de tiempo fue el que se podían utilizar Templates de Bootstrap para realizar la práctica, por lo que no perdimos tanto tiempo en la realización de las interfaces utilizadas. De todas formas, a pesar de las dificultades que tuvimos, la practica fue interesante y pudimos ver los conceptos de clase en acción con más claridad, por lo que consideramos que fue bastante productivo lo que se obtuvo de esta.

## Bibliografía

1. Quickstart. (n.d.). Retrieved June 03, 2016, from <http://www.h2database.com/html/quickstart.html>
2. If, else, elseif. (n.d.). Retrieved June 03, 2016, from <http://freemarker.org/docs/ref_directive_if.html>
3. CSS. (n.d.). Retrieved June 03, 2016, from <http://getbootstrap.com/css/>
4. Start Bootstrap Blog. (n.d.). Retrieved June 06, 2016, from <http://blog.startbootstrap.com/>