Arquitecturas para Aplicaciones en Red (2021-2022)

Práctica 2 - Web Sockets & API REST

Introducción

En esta práctica se desarrollará un sistema distribuido que consiste en una **aplicación basada en websockets y un webservice de tipo REST** que permite enviar, ver y gestionar información en tiempo real sobre el precio actual en el cambio de monedas (dólar, euro, bitcoin, ethereum, etc.) a los exploradores web que se suscriban al servicio.

Aplicación WebSockets

La funcionalidad de la aplicación basada en WebSockets seguirá el modelo **publicación-suscripción** bajo el cual **el servidor enviará regularmente a los clientes información actualizada** sobre el precio de intercambio de un conjunto limitado de divisas. Los clientes podrán elegir las parejas de divisas de las cuales quieren conocer el precio. Todos los clientes recibirán la misma información, esto es, si un cliente pide una determinada pareja de divisas, a partir de ese momento todos los clientes recibirán la información de dicha pareja.

La aplicación se desplegará en un servidor web **Apache Tomcat 9**.

La **funcionalidad que se debe ofrecer al usuario** es la siguiente:

- El usuario podrá elegir un conjunto de parejas de monedas de las cuales quiere realizar el seguimiento de su precio de cambio. Esta información se enviará al servidor y a partir de ese momento publicará la información correspondiente de forma periódica a todos los clientes que estén suscritos.
- El usuario podrá eliminar las parejas de monedas de las cuales ya no quiera conocer el precio.
- Se mostrará al usuario el precio actual de cambio para cada pareja seleccionada y la hora de la última actualización de dicho precio.

Se da libertad sobre cómo implementar la interfaz del usuario para permitir las tres operaciones anteriores, aunque siempre ha de ser una interfaz que funcione en un navegador web. Una opción sería poder añadir una pareja de monedas cada vez y, de la misma manera, poder eliminar las parejas de las cuales se recibe información una a una. En cualquier caso, las acciones que realice un cliente tendrán efecto en la información que reciben todos los demás.

Respecto al layout en sí, lo normal sería tener una sección de la web en la cual el usuario pueda enviar sus selecciones al servidor y otra sección en la que se muestran los precios de

las parejas de divisas seleccionadas. **La idoneidad de la solución aplicada tendrá efecto en la nota**.

El servidor será el encargado de publicar la información de las parejas de monedas solicitadas, aunque no es quien tiene la información. Cualquier información la tendrá que pedir al webservice REST, la aplicación de WebSockets **no debe guardar nada acerca de los intercambios.** Esto significa que no debe saber qué monedas están disponibles para escoger, cuáles son los intercambios activos, cuál es el valor máximo de cada intercambio, etc. Toda esta información se tendrá que ir a preguntar al webservice REST.

El servidor publicará la información respecto a cada pareja seleccionada por los usuarios según **el intervalo de tiempo fijado por una variable local**. La idea en cualquier caso es publicar nueva información cada minuto; pero para hacer pruebas puede ser interesante actualizar los datos cada 5 o 10 segundos, etc. Para que se siga correctamente el modelo publicación-suscripción, es importante que no sea el propio navegador web quien haga las peticiones al servidor cada x segundos, esto debe ser gestionado por el servidor. Se recomienda el uso de un Thread que tenga acceso a todas las sesiones activas.

Podemos resumir la actuación del servidor como un proceso que:

- Gestiona las sesiones activas (los usuarios que estén suscritos).
- Cada x segundos, realiza una petición (o varias peticiones) al servidor REST para recibir los valores de los intercambios actualizados y pasarlos a los usuarios conectados.
- Cuando un nuevo usuario se conecta a la aplicación, debe gestionar la sesión y mostrar toda la información actual (**sin pedir una actualización de los intercambios)** al usuario que se acaba de conectar.

API REST

Se trata de **un webservice de tipo REST** basado en las librerías JERSEY (JAX-RS) el cual permitirá conocer información sobre el precio en el cambio de monedas (dólar, euro, bitcoin, ethereum, etc.).

Dicho webservice se desplegará sobre un servidor web Apache Tomcat 9 dedicado en exclusiva a esta aplicación (no se debe reutilizar el mismo Apache Tomcat dónde se ejecuta el servidor de WebSockets).

Las operaciones que el webservice debe ofrecer son las siguientes:

Verbo	Recurso	Descripción
`GET`	`/currencies`	Listar todas las monedas que actualmente el servicio tiene almacenadas. Un usuario de la aplicación de WebSockets sólo debe poder

		suscribirse a un intercambio formado a partir de estas monedas.
POST	/currencies	Se crea una moneda nueva. Los datos de la moneda se pasan por el BODY del HTTP request.
DELETE	/currencies/{id}	Se elimina una moneda. Al eliminarse esta moneda, también se deben eliminar todos los intercambios que usen esta moneda.
GET	/exchange-rates	Listar todas las parejas de divisas que actualmente el servicio tiene almacenadas, junto con su precio actual, su precio máximo, la fecha/hora de la consulta, y la fecha/hora del precio máximo.
GET	/exchange-rates/current/{id}	Mostrar el precio actual de la pareja de divisas indicada según el identificador, y la fecha/hora de la consulta.
GET	/exchange-rates/max/{id}	Mostrar el precio máximo de la pareja de divisas en todo el tiempo en el que se le ha realizado el seguimiento, indicada según el id, y mostrando la fecha/hora asociada a ese valor máximo.
POST	/exchange-rates	Se crea una pareja de divisas en el servidor sobre la cual habrá que realizar un seguimiento de cotizaciones. Los datos de las divisas se pasan por el BODY del HTTP request.
DELETE	/exchange-rates/{id}	Se elimina la pareja de divisas en el servidor sobre la cual se estaba realizando el seguimiento.

Todas las respuestas deben usar el **formato JSON**.

El webservice obtendrá los datos reales de cotización de intercambio de divisas consumiendo a su vez el webservice ofrecido por Cryptonator. Tendréis que hacer peticiones HTTP a esta API para obtener la información del intercambio correspondiente. Por ejemplo, una petición HTTP a la URL https://api.cryptonator.com/api/ticker/btc-usd nos proporcionaría con la siguiente información. Podéis probarlo haciendo click directamente en el enlace, abriéndolo con el navegador:


```
{
    "ticker": {
        "base": "BTC",
        "target": "USD",
```

```
        "price": "64117.54929778",
        "volume": "27025.15502514",
        "change": "203.91715282"
    },
    "timestamp": 1636918922,
    "success": true,
    "error": ""
}
```


El webservice que se debe implementar, **solo ofrecerá los datos sobre las parejas de divisas sobre las cuales se esté realizando el seguimiento**. Cada vez que se le pida a dicho webservice el valor actual de cotización de una pareja en concreto, **el webservice consultará y actualizará si es necesario el valor máximo de cotización obtenido**. Es necesario registrar la fecha/hora en el cual se obtuvo el valor máximo de una cotización.

Los datos almacenados, tanto el valor actual como el valor máximo de una pareja, no se deben guardar en variables de sesión, sino que **se debe usar la base de datos H2** (de la misma manera que se usa en el proyecto de ejemplo). Esto implica **seguir el patrón DAO** y usar las anotaciones necesarias en las entidades creadas.

Entrega

La práctica se realizará de forma individual o por parejas. Se pide la **carpeta de las aplicaciones** con el código fuente y los binarios. También será necesario **un fichero** (PDF) que indique el nombre del alumno, su dirección de correo electrónico y las decisiones de diseño tomadas durante la práctica junto a su justificación.

Utilizad un ZIP (Prac2_ApellidoAlumno1ApellidoAlumno2.zip) para agrupar y comprimir el directorio y el documento solicitado. La entrega de la práctica se realizará utilizando el Moodle adjuntando dicho zip a la tarea Práctica 2 – WebSockets & API REST.

El límite de entrega es el **viernes 7 de enero del 2022 a las 23:59**.

Proceso de corrección

La corrección se realizará mediante una entrevista en las horas del laboratorio de los días **11 y 12 de enero**. Principalmente se probará el funcionamiento de la aplicación (con varios ordenadores si hiciera falta) y el grado de conocimiento del trabajo realizado.