Universidad Nacional de Lanús

DESARROLLO DE SOFTWARE EN SISTEMAS DISTRIBUIDOS

Mensajería con Apache Kafka

Tomando como base el sistema creado en el trabajo práctico de RPC, desarrollar y modificar las funcionalidades detalladas en el apartado de requerimientos, utilizando **Apache Kafka** para producir y consumir los mensajes de los topics.

Requerimientos

1. **Alta de receta:** cuando un usuario publica una receta, se deberá guardar en un topic de Kafka "Novedades", la siguiente información: nombre de usuario, titulo de la receta, url de la primera foto.

2. **Página de inicio:** al entrar al sistema, tiene que mostrarse una página de bienvenida que muestra las 5 últimas recetas creadas, de las cuáles se tomará la información consumiendo el topic de Kafka mencionado en el punto anterior. Esta página de inicio también contará con secciones para ver las recetas y usuarios más populares. Estos últimos no se consumen de ningún topic, sino de los datos que persista el consumidor de popularidad (punto 5).

3. **Usuarios:** en la funcionalidad para seguirlo, se guarda en el topic "PopularidadUsuario": *nombreUsuario*, *puntaje* (+1) si se comienza a seguir o (-1) si se deja de seguir.

4. **Recetas**: cuando un usuario entre a una receta, además de poder guardarla como favorita, también podrá comentarla y/o calificarla (de 1 a 5 estrellas).

    a. **Comentario:** esta acción no persiste en la base de datos directamente, sino que guardará en un topic "Comentarios" la información: *usuarioComentario, recetaComentada, comentario.* Los comentarios se guardarán en la base mediante un consumidor (punto 6).

    También guardará en un topic "PopularidadReceta" de Kafka la siguiente información: *id de la receta*, *puntaje* (+1).

    b. **Calificación:** también se guardará la información en el mismo topic, pero el puntaje corresponderá a las estrellas que el usuario le dio a la receta.

    c. **Favorita:** guarda la información en el topic "PopularidadReceta", pero *puntaje* (+1) si la marcó como favorita o con (-1) si la desmarcó. Además, aplica el mismo criterio pero guardando en el topic "PopularidadUsuario": *nombreUsuario* del creador de la receta, *puntaje* (+-1) según el caso.

    d. **Visualización:** agregar las modificaciones necesarias para mostrar el puntaje promedio de la receta y los comentarios de la misma.

    e. **Restricciones:** el usuario que creó la receta, no puede calificar ni marcar como favorita su propia receta. Puede comentar, pero su comentario no cuenta para el cálculo de popularidad.

5. **Popularidad:**
    a. Receta: mediante un proceso programado que se ejecutará automáticamente cada un tiempo configurable (por ejemplo, cada 1 hora), se consumirá la información almacenada en el topic de "PopularidadReceta", el cual hará el cálculo de popularidad con esos datos junto a los existentes en la base de datos. Luego de realizar los cálculos pertinentes, guardará esos resultados en la base de datos.

b. **Usuarios:** igual al anterior, pero para el topic “PopularidadUsuario” y en un proceso aparte.

6. **Persistencia de comentarios:** similar al punto anterior, habrá un proceso programado que consuma los mensajes del topic “Comentarios” y guardará en forma masiva todos esos comentarios.

NORMAS DE ENTREGA

El trabajo entregado deberá contener un documento incluyendo:

- La estrategia de resolución del trabajo práctico. Es un texto descriptivo de cómo se estructuró la aplicación, todo aquello que consideren significativo para explicar la resolución del trabajo: diagrama del modelo de datos, arquitectura del sistema, etc.
- El código fuente, **DE PROPIA AUTORÍA**, del proyecto subido a un repositorio público de **Github**.
- Las pruebas realizadas con las respectivas capturas de pantalla.
- Integrantes del grupo y las tareas realizadas por cada uno.

El incumplimiento de cualquiera de las normas de entrega implicará la desaprobación del trabajo práctico.