Universidad de los Andes
Ingeniería de Sistemas y Computación
ISIS1206 - Estructuras de Datos
Ejercicio N17 – QueVideo 2.0


Cupi2

Problema

Después de su gran trabajo, QueVideo tiene un flujo permanente de visitas y el número de usuarios va en aumento. Se hace cada vez más importante que los diferentes usuarios puedan interactuar entre sí. Se requiere entonces que usted adicione nuevas funcionalidades a su plataforma, para soportar el proceso de "socialización" de sus usuarios y ayudar a realizar búsquedas y recomendaciones de videos más efectivas.

Se requiere el concepto de Video Contacto, definido como un usuario registrado de la plataforma con el que otro usuario determinado comparte información. La relación es unidireccional, de modo que cuando un usuario *X* añade como Video Contacto a un usuario *Y*, el usuario *Y* no añade automáticamente al usuario *X* como Video Contacto. En la relación de Video Contacto, el usuario *X* se interesa por ser seguidor del usuario Y, lo cual le permitiría al usuario *X* conocer la información pública del usuario Y. Dadas las nuevas características de la plataforma, se requiere que cuando un usuario se registre, para cada dato proporcionado indique si este es público o no. Al realizar búsquedas y consultar detalles de usuario, solo los datos públicos (aquellos que cada usuario halla autorizado) deben ser mostrados. Adicionalmente, un usuario también puede indicar si quiere poder ser seguido libremente o si desea aprobar cada petición de seguimiento.

Aprovechando la red de usuarios que se conformará, es necesario definir heurísticas para recomendar videos y nuevos usuarios.

Finalmente, para asegurarse de que la iniciativa social dentro de la plataforma esta funcionando como se espera, la administración debe mostrar nuevas estadísticas sobre el comportamiento de los usuarios.

Requerimientos funcionales

Su aplicación Web debe soportar los siguientes requerimientos funcionales:

Requerimientos de Usuario

RU1. Buscar a un usuario por su nombre o correo electrónico. Se debe mostrar la lista de todos los usuarios que contengan la cadena de búsqueda en su nombre o en su correo electrónico (según lo el criterio de búsqueda).

RU2. Ver el detalle de un usuario. Se deben poder consultar sus datos públicos y además, si hay conexión entre el usuario actual y el usuario buscado. La conexión del usuario actual con un usuario buscado se da si hay al menos un camino entre el usuario actual y el usuario buscado en

cualquiera de los dos sentidos (*X* es seguidor directo o transitivo de *Y* o *X* es seguido directamente o indirectamente por *Y*). Se debe mostrar el camino más corto entre los dos usuarios y precisar si el usuario es seguidor directo/transitivo del usuario buscado o si es un usuario seguido por el usuario buscado.

RU2. **Adicionar a un usuario como Video Contacto.**

RU3. **Aceptar a un usuario como Video Contacto.**

RU4. Dado un video, consultar cuáles de los Video Contactos de un usuario le han dado like.

RU5. Sugerir Video Contactos. Con base en una heurística definida por usted, mostrar una lista (puede ser de una longitud determinada) de sugerencias de usuarios a contactar.

RU6. Sugerir videos a un usuario. Con base en los videos vistos e/o indexados por sus Video Contactos, se deben sugerir videos al usuario conectado, teniendo en cuenta etiquetas, número de veces visto, etc.

Requerimientos de Administrador

RA1. Consultar todos los usuarios *influencer*. Buscar a aquellos usuarios que no siguen a nadie, pero son seguidos.

RA2. Consultar todos los usuarios *stalker.* Buscar a aquellos usuarios que no son seguidos, pero siguen a otros usuarios.

RA3. Consultar todos los usuarios *foreverAlone.* Buscar a aquellos usuarios que no son seguidos ni siguen a nadie.

RA4. Consultar el camino mas largo que se forma en la red de usuarios. Es necesario mostrar las personas que conforman tal camino.

RA5. Consultar al usuario con más contactos. Buscar aquel cuya suma de Video Contactos que sigue y que lo siguen es máxima en la red de usuarios.

RA6. Buscar un usuario *Hamilton*. Buscar a aquel usuario desde él que se puede llegar a todos los usuarios, formando un camino Hamiltoniano.

Restricciones de diseño

1. El diseño se debe hacer **MINIMIZANDO el tiempo de ejecución de todas las operaciones y el espacio que ocupan las estructuras de datos, garantizando que el programa pueda evolucionar**.

2. En particular, se deben justificar las decisiones de diseño tomadas, se debe explicar claramente en un **documento de diseño** por qué se escogieron las estructuras de datos utilizadas, teniendo en cuenta los requerimientos descritos anteriormente. Adicionalmente, es

importante que en este documento (y definido desde antes de la implementación) esté el mapa del sitio, en este se debe indicar cuáles páginas existen, desde qué página se posible navegar a cuáles y cuáles son las funcionalidades proporcionadas en cada una de ellas.

3. El sistema debe ser **persistente**. Recuerde que el objetivo del ejercicio es resolver los requerimientos con las estructuras en memoria principal, por este motivo **no se acepta el uso de bases de datos**. Puede utilizar serialización para persistir el modelo del mundo.

4. El sistema debe ser desarrollado como una aplicación WEB. Para ello, debe definir su modelo del mundo como un singleton, un conjunto de páginas estáticas y un conjunto de páginas dinámicas.

5. La interfaz gráfica **debe ser implementada utilizando servlets** y ser usable y amigable. Se debe tener una navegación clara (siempre poder volver a la página de inicio) y utilizar correctamente el diseño, la tipografía y los colores para presentar los diferentes contenidos.

6. El diseño debe estar desacoplado con interfaces.

7. Debe implementar las estructuras de datos **genéricas** en un **paquete aparte**. Las estructuras de datos se deben probar con **pruebas unitarias**.

8. Los requerimientos deben resolverse basados en una estructura de datos Grafo Dirigido.

9. El sistema debe utilizar **ant** para su compilación y ejecución.

10. No se puede utilizar Java Collections ni Cupi2 Collections.

11. No debe haber regresiones de software. Es decir, todas las funcionalidades requeridas para la primera versión de QueVideo, deben seguir funcionando correctamente en la nueva versión.

Entregables

Entrega Avance (30%) Lunes 25 Nov (12:00 m)
E1. Diagrama de clases del Diseño de la EDT Grafo Dirigido
E2. Proyecto Eclipse con Implementacion y Pruebas de la EDT Grafo Dirigido (incluye documento con la ilustración de los grafos dirigidos de prueba).
E3. Documento de planeación y tiempos (Definición, priorización, planeación de tareas, estimación de tiempos y tiempos efectivos para la entrega de avance).

Entrega Única (70%) Jueves 28 Nov (7:00 am)
E1. Uso de ant para generación de libreria .jar de Estructura de Datos y para la compilación y despliegue de la aplicación web.
E2. Documento de diseño que incluya:
- > Análisis.
- > Diseño completo de la aplicación.
- > Justificación de la existencia de entidades definidas.
- > Estructuras de datos seleccionadas.
- > Justificación de la selección de estructuras de datos.

> Complejidades de cada uno de los requerimientos funcionales.

> Mapa de navegación definitivo.

E3. Implementación completa de las Estructuras de Datos genéricas (en un paquete aparte), incluyendo sus pruebas unitarias completas (si hubo algún cambio desde la Entrega 1).

E4. Implementación completa de la interfaz web.

E5. Implementación completa de los requerimientos. Se califican únicamente aquellos requerimientos que sean funcionales desde la interfaz gráfica. Los requerimientos deben funcionar ejecutándose desde la aplicación web.

E6. Javadoc de la aplicación.

E7. Documento de planeación y tiempos (Definición, priorización, planeación de tareas, estimación de tiempos, tiempos efectivos para cada entrega y análisis de la planeación y ejecución) – versión completa

---

El mejor ejercicio de cada sección tendrá un bono especial sobre la nota del nivel.