UNIVERSIDAD NACIONAL DE SAN MARTÍN

Programación con Herramientas Modernas

Trabajo práctico 2024

Dominio y precondiciones

El dominio sigue siendo el mismo de la entrega 0. Asumimos que están persistidos los objetos de dominio en una base de datos relacional + una documental.

Parte 1: Persistencia clave-valor

A partir de ahora vamos a disponer de una tercera alternativa de persistencia con un motor de base de datos no relacional clave/valor. Se deberá persistir el carrito de compras guardado el resultado de una búsqueda con un *time to live* de 18.000 segundos (5 horas). Nuestra recomendación es utilizar Redis con la persistencia que provee Spring Boot o bien con el driver Jedis.

Parte 2: Persistencia en grafos

Necesitamos persistir al usuario y otros elementos necesarios en una base de datos de grafos. El principal motivo es realizar las sugerencias de amigos teniendo en cuenta a la persona y sus compras. En la búsqueda para agregar amigos en el perfil de usuario tiene que mostrar los amigos de los amigos del usuario que hayan concurrido a los mismos shows.

Además en base al modelado propuesto cada integrante deberá resolver las siguientes consultas ***desde el IDE de la base de datos*** utilizando el lenguaje de la base de datos seleccionada:

- Saber qué usuarios compraron tickets para un determinado show y fecha.
- Saber qué usuarios tienen más de 4 tickets.
- Saber una sugerencia de shows que podrían interesar. Son los shows que fueron amigos del usuario con los que compartieron entradas pero el usuario no asistió.
- Saber qué shows tienen más de 4 reservas.
- Saber qué usuario tiene un ticket con un show que tiene más de 50 tickets.

Nuestra sugerencia es elegir Neo4J (community server) como base de grafos.

Puede ver en el sitio web de la materia cómo se instala cada una de las tecnologías.

Persistiendo la misma entidad en diferentes repositorios

Si intentamos persistir una entidad mapeada en más de un repositorio, nos encontramos con el problema que Spring tiene un único mapper de objetos que relaciona a una clase con un repositorio. Si hay más de un repositorio por clase, debemos especificar contra qué tipo de base de datos debe persistir. Esto se logra con ciertas annotations en particular que se definen en la clase que contiene el **main**. Las mismas son

```
@EnableMongoRepositories(basePackageClasses = NOMBRE_DELREPOSITORIO)
@EnableJpaRepositories (basePackageClasses = NOMBRE_DELREPOSITORIO)
@EnableRedisRepositories (basePackageClasses = NOMBRE_DELREPOSITORIO)
@EnableNeo4jRepositories (basePackageClasses = NOMBRE_DELREPOSITORIO)
```

Responsabilidad grupal

- Revisión del modelo de objetos, haciendo ajustes que sean necesarios
- Definición de los componentes de acceso a datos

Cada integrante debe ser responsable de encontrar los componentes adecuados para cada caso de uso asignado: application model si correspondiere, objetos de dominio que colaboran, controllers, services y vista.

En caso de haber menos integrantes se conversará con el tutor la funcionalidad del TP a entregar.