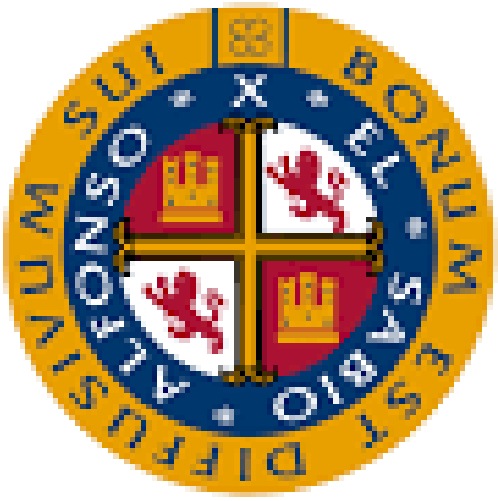

UNIVERSIDAD
ALFONSO X EL SABIO

SPRING BATCH en Java

Practica 3

By

Luis Miguel Urbez Villar

NP: 139314

Programación concurrente Ingeniería Informática

Universidad Alfonso X el Sabio

RESUMEN

El proyecto es una aplicación web desarrollada en Java utilizando el marco de trabajo Spring Boot. La aplicación utiliza Spring Batch para procesar datos de usuarios, que se leen desde un archivo CSV y se escriben en una base de datos.

El controlador ‘LogStreamController‘ maneja las solicitudes GET para importar clientes y transmitir registros. Utiliza un servicio de ejecución no bloqueante para manejar las tareas en segundo plano. Este controlador también tiene un método ‘streamLogs‘ que transmite los registros de la aplicación en tiempo real utilizando Server-Sent Events (SSE).

El controlador ‘JobController‘ maneja las solicitudes GET y POST para trabajos y la carga de archivos. Este controlador utiliza ‘JobLauncher‘ para lanzar trabajos de Spring Batch. Cuando se sube un archivo, se guarda en la carpeta de recursos del proyecto y luego se lanza un trabajo de Spring Batch para procesar los datos del archivo.

La configuración de Spring Batch se encuentra en el archivo ‘SpringBatchConfig.java‘. Define un ‘Job‘ que procesa datos de usuarios desde un archivo CSV. El ‘Job‘ se divide en pasos, cada uno de los cuales consta de un lector, un procesador y un escritor. El lector lee los datos del archivo CSV, el procesador procesa los datos y el escritor los escribe en una base de datos.

El proyecto también incluye una clase de dominio ‘Usuario‘ y un repositorio ‘UsuarioRepository‘ para interactuar con la base de datos. La clase ‘Usuario‘ representa a un usuario en la base de datos, y ‘UsuarioRepository‘ proporciona métodos para realizar operaciones CRUD en la tabla de usuarios en la base de datos.

En resumen, este proyecto es una aplicación web que utiliza Spring Boot y Spring Batch para leer, procesar y escribir datos de usuarios desde un archivo CSV a una base

de datos.

_____

YOUR NAME

SPRING BATCH

Spring Batch es un marco de trabajo que se utiliza para procesar grandes volúmenes de datos de manera eficiente. En este proyecto, se utiliza para leer datos de usuarios desde un archivo CSV y escribirlos en una base de datos.

La configuración de Spring Batch se encuentra en el archivo ‘SpringBatchConfig.java‘. Aquí se definen varios beans que son necesarios para el funcionamiento de Spring Batch.

El bean ‘reader‘ es un ‘FlatFileItemReader<Usuario>‘. Este lector se configura para leer desde un archivo CSV en el directorio de recursos del proyecto. Utiliza un ‘DefaultLineMapper<Usuario>‘ para mapear cada línea del archivo CSV a un objeto ‘Usuario‘. El ‘DelimitedLineTokenizer‘ se utiliza para dividir cada línea en campos basándose en un delimitador, que en este caso es una coma. Luego, cada campo se asigna a una propiedad del objeto ‘Usuario‘ utilizando un ‘BeanWrapperFieldSetMapper<Usuario>‘.

El bean ‘processor‘ es un ‘UsuarioProcessor‘. Este procesador se utiliza para procesar cada objeto ‘Usuario‘ después de que se ha leído del archivo CSV. En este caso, el procesador no se ha definido en el código proporcionado, por lo que no se puede determinar qué procesamiento se realiza.

El bean ‘writer‘ es un ‘JdbcBatchItemWriter<Usuario>‘. Este escritor se configura para escribir cada objeto ‘Usuario‘ procesado en una base de datos. Utiliza un ‘BeanPropertyItemSqlParameterSourceProvider<>‘ para mapear las propiedades del objeto ‘Usuario‘ a los parámetros de la consulta SQL. La consulta SQL para insertar un usuario en la base de datos se define en el método ‘writer‘.

El bean ‘job‘ es un ‘Job‘. Este trabajo se configura para ejecutar un solo paso, que se define en el bean ‘step1‘. El paso se configura para leer, procesar y escribir objetos ‘Usuario‘ en lotes de 100. Utiliza el lector, el procesador y el escritor definidos

anteriormente. También se configura para utilizar un ‘TaskExecutor‘ para ejecutar tareas en paralelo.

En resumen, la configuración de Spring Batch en este proyecto lee datos de usuarios desde un archivo CSV, procesa los datos y los escribe en una base de datos en lotes de 100 utilizando múltiples hilos.

CONCURRENCIA USADA

En el proyecto, la concurrencia se maneja principalmente a través del uso de Spring Batch y un ‘TaskExecutor‘.

Spring Batch es un marco de trabajo que permite el procesamiento de datos en lotes. En tu proyecto, se utiliza para leer datos de un archivo CSV, procesarlos y escribirlos en una base de datos. Este proceso se realiza en lotes de 100 elementos a la vez, lo que permite un procesamiento más eficiente de los datos.

El ‘TaskExecutor‘ es donde realmente se maneja la concurrencia. En tu configuración de Spring Batch, has definido un ‘TaskExecutor‘ en el método ‘taskExecutor()‘. Este ‘TaskExecutor‘ es un ‘ThreadPoolTaskExecutor‘, que es una implementación de ‘TaskExecutor‘ que utiliza un grupo de hilos para ejecutar tareas.

El ‘ThreadPoolTaskExecutor‘ se configura con un tamaño de pool de hilos de 200, tanto para el pool de hilos principal (‘setCorePoolSize(200)‘) como para el pool de hilos máximo (‘setMaxPoolSize(200)‘). Esto significa que puede manejar hasta 200 tareas concurrentes. Después de configurar el tamaño del pool de hilos, se llama al método ‘afterPropertiesSet()‘ para inicializar el ‘TaskExecutor‘.

Este ‘TaskExecutor‘ se utiliza en el método ‘step1()‘, donde se configura el paso de Spring Batch. El paso se configura para leer, procesar y escribir objetos ‘Usuario‘ en lotes de 100. El ‘TaskExecutor‘ se pasa al método ‘taskExecutor()‘ del constructor del paso, lo que significa que este paso se ejecutará utilizando el ‘TaskExecutor‘ y, por lo tanto, se beneficiará de la concurrencia.

En resumen, tu proyecto utiliza Spring Batch para el procesamiento de datos en lotes y un ‘ThreadPoolTaskExecutor‘ para manejar la concurrencia. Esto permite que tu aplicación procese grandes cantidades de datos de manera eficiente y concurrente.

CONCLUSIÓN

A través de este proyecto, has adquirido una comprensión sólida de varios conceptos y tecnologías clave en el desarrollo de aplicaciones Java con Spring Boot. Aquí hay un resumen de lo que has aprendido:

1. Spring Boot: Has utilizado Spring Boot para crear una aplicación web robusta y eficiente. Spring Boot facilita la creación de aplicaciones independientes basadas en Spring que puedes "simplemente ejecutar".

2. Spring Batch: Has aprendido a utilizar Spring Batch para el procesamiento de datos en lotes. En tu proyecto, se utiliza para leer datos de un archivo CSV, procesarlos y escribirlos en una base de datos.

3. Concurrencia: Has aprendido a manejar la concurrencia en tu aplicación utilizando un ‘TaskExecutor‘. En tu proyecto, has utilizado un ‘ThreadPoolTaskExecutor‘ para manejar hasta 200 tareas concurrentes.

4. Lectura y escritura de archivos CSV: Has aprendido a leer datos de un archivo CSV utilizando un ‘FlatFileItemReader‘ y a escribir datos en una base de datos utilizando un ‘JdbcBatchItemWriter‘.

5. Mapeo de datos: Has aprendido a mapear datos de un archivo CSV a objetos ‘Usuario‘ utilizando un ‘DefaultLineMapper‘, un ‘DelimitedLineTokenizer‘ y un ‘BeanWrapperFieldSetMapper‘.

6. Gestión de dependencias: Has utilizado Maven para la gestión de dependencias en tu proyecto Java y NPM para el manejo de paquetes de JavaScript.

7. Controladores en Spring: Has aprendido a crear controladores en Spring para manejar las solicitudes HTTP y a utilizar servicios para manejar la lógica de negocio.

8. Interacción con la base de datos: Has aprendido a interactuar con la base de datos utilizando un repositorio, en este caso, ‘UsuarioRepository‘.

En resumen, este proyecto te ha proporcionado una experiencia práctica valiosa en

el desarrollo de aplicaciones web con Spring Boot, el procesamiento de datos en lotes con Spring Batch, la gestión de la concurrencia, la lectura y escritura de archivos CSV, el mapeo de datos, la gestión de dependencias y la interacción con la base de datos.