	SOFKA		Assessment Technical	
	Nº Versión: 1	Fecha: 15-07-2024	Página 1 de 2	

Prueba Técnica: Sistema de Procesamiento de Transacciones

Objetivo

Trabajamos para una empresa de transporte masivo, se desea diseñar y desarrollar un sistema que reciba y procese transacciones de los usuarios a través de una API REST. El sistema debe almacenar las transacciones en MongoDB, registrar cada transacción en un broker de mensajería y generar un resumen diario del valor total de las transacciones.

Requerimientos

1. Tecnologías:


- Java (Spring Boot o Quarkus)
- MongoDB
- Docker
- Broker de mensajería (por ejemplo, ActiveMQ, RabbitMQ, Kafka)

2. Funcionalidades:

- La API debe recibir solicitudes POST con un objeto de transacción que contenga los siguientes campos:
 - **transactionId** (ID de la transacción)
 - **timestamp** (Fecha y hora)
 - **deviceNumber** (Número de dispositivo)
 - **userId** (Número de usuario)
 - **geoPosition** (Geoposición de la transacción)
 - **amount** (Valor de la transacción)
- El sistema debe ser capaz de procesar hasta 10,000 transacciones por minuto y procesar cada una en menos de 100 ms.
- Las transacciones deben almacenarse en MongoDB.
- Se debe registrar cada transacción en el broker de mensajería.
- Generar un documento en MongoDB con el valor total que contenga la suma del valor de las transacciones, agrupadas por día.

3. Consideraciones:

- La configuración de MongoDB y el broker de mensajería debe realizarse utilizando Docker.
- Estrategias de alta disponibilidad y recuperación ante fallos no son necesarias para esta prueba, pero serán valoradas si se incluyen.
- Puedes agregar cualquier consideración o asumir consideraciones que no estén especificadas, en caso de ser necesario, por favor documentarlas.

	SOFKA		Assessment Technical	
	Nº Versión: 1	Fecha: 15-07-2024	Página 2 de 2	

Entregables

1. Código Fuente:

- Todo el código fuente del proyecto debe estar incluido.
- Utilizar un repositorio público de Git (por ejemplo, GitHub).

2. Guía de Ejecución:

- Instrucciones claras para ejecutar la aplicación.
- Detalles sobre cómo configurar y ejecutar los contenedores Docker.
- Ejemplos de cómo enviar solicitudes POST a la API.

3. Documentación:

- **Consideraciones asumidas:** Cualquier suposición hecha durante el desarrollo del sistema.
- **Estrategias Utilizadas:** Una breve explicación de las estrategias y patrones utilizados para cumplir con los requisitos del sistema.
- **Guía de Uso:** Cómo utilizar y probar el sistema.

Evaluación

- **Funcionalidad:** Verificar que la API recibe y procesa las transacciones correctamente.
- **Rendimiento:** Evaluar si el sistema puede manejar 10,000 transacciones por minuto.
- **Documentación:** Calidad y claridad de la guía de ejecución y la documentación de estrategias.
- **Código:** Calidad y buenas prácticas en el código fuente.

Instrucciones para la Entrega

1. **Repositorio Git:** Crear un repositorio público en GitHub con el código fuente del proyecto.
2. **Docker Compose:** Incluir un archivo `docker-compose.yml` para configurar y ejecutar MongoDB y el broker de mensajería.
3. **Procesamiento y Almacenamiento:** Asegurar que las transacciones se procesen en el tiempo estipulado y se almacenen correctamente en MongoDB.
4. **Registro en el Broker:** Implementar el registro de cada transacción en el broker de mensajería.
5. **Resumen Diario:** Implementar la generación de un documento diario en MongoDB con el valor total de las transacciones.