Microservicio de Conversión de Costos de Vehículo a Criptomoneda con Spring WebFlux

Descripción General

Desarrollo de una API RESTful que permita convertir el precio de distintos modelos de vehículos, expresado en dólares, a su equivalente en distintas criptomonedas.

Los requerimientos funcionales y técnicos se presentan a continuación.

El ejercicio se debe realizar implementando la mayor y mejores prácticas relacionadas a programación funcional y Spring Webflux, en caso de que el ejercicio no utilice programación reactiva, no se tomará en cuenta el ejercicio para la evaluación.

Complejidad por seniority

Nota: Considerar las siguientes indicaciones en base al perfil al que está aplicando. Ejemplo: si es un perfil semisenior sólo realizar lo indicado para este perfil.

SemiSenior: Desarrollo de microservicios. R1, R2, R3, R5, R7, R8, R9, R10.

Senior: Desarrollo de microservicios. R1, R1.1, R2, R3, R5, R6, R7, R8, R9, R10, R11, R12.

Herramientas

- Deberás utilizar Java 17, Spring Boot y Spring WebFlux para el desarrollo del microservicio.

Requerimientos

R1 **API de Conversión:** Proporcionará una lista de versiones para un modelo específico de vehículo, mostrando el nombre de la versión, el año del vehículo y su precio tanto en dólares como en la criptomoneda seleccionada. Las criptomonedas admitidas para la conversión son Bitcoin (BTC), Ethereum (ETH). Persistir la cotización.

R1.1 Implemente un control de fallo, para que en caso de que el API principal no se encuentre disponible, el API secundario sea utilizado. En caso de que el cliente especifique otra criptomoneda no soportada, se debe mostrar un mensaje de error.

APIs a consumir:

Vehículos

- ALL NEW ACCENT -> https://kerner.hyundai.com.ec/api/versiones/ 1/1036
- ALL NEW TUCSON -> https://kerner.hyundai.com.ec/api/versiones/ 1/1031
- SANTA FE -> https://kerner.hyundai.com.ec/api/versiones/1/1023

- GRAND i10 -> https://kerner.hyundai.com.ec/api/versiones/1/1038

Cotización de criptomoneda

BTC

- principal https://http-api.livecoinwatch.com/coins/BTC/about?currency=USD

El valor de la cotización se encuentra en data.lastPrice (precio de 1 bitcoin en dólares)

- secundaria https://api.coinlore.net/api/ticker/?id=90

  El valor de cotización se encuentra en price_usd (precio de 1 bitcoin en dólares)

ETH

- principal https://http-api.livecoinwatch.com/coins/ETH/about?currency=USD

El valor de la cotización se encuentra en data.lastPrice (precio de 1 bitcoin en dólares)

- secundaria https://api.coinlore.net/api/ticker/?id=80

  El valor de cotización se encuentra en price_usd (precio de 1 bitcoin en dólares)

Ejemplo API conversión

API de conversion

Entrada

```
{
"data": {
       "model": "TUCSON",
       "cryptocurrency": "BTC"
}
}
```

Salida

```
{
 "data": {
       "convertionId": "xxx-xxx-xxx-xxxx", // id generado

       "conversionTimelife": "20 seconds", // tiempo restante que se
       mantiene el precio de la conversión para perfil senior, usted define el
       tiempo
       "versions": [
        {
        "model": "TUCSON", // dato de API vehiculos
        "version": "TUCSON TL", // dato de API vehiculos
```

```
        "priceUsd": 35000.0, // dato de API vehiculos
        "priceCryptocurrency": 1.6, // valor precio vehiculo / valor cripto
        "cryptocurrency": "BTC" // criptomoneda con la que se cotizó
        },{
        "model": "TUCSON",
        "version": "TUCSON SJ",
        "priceUsd": 25000.0,
        "priceCryptocurrency": 1,
        "cryptocurrency": "BTC"
        }
      ]
 }
}
```

R2 **API de Compra:** Permitirá la realización de una compra utilizando la conversión de criptomoneda generada. La información de la compra (la conversión, el nombre del cliente y la fecha de compra) se almacenará de manera persistente tal como se especifica en los requerimientos R5 o R6 dependiendo del nivel de seniority.

Ejemplo API Compra

Entrada

```
{
 "data": {
       "convertionId": "xxx-xxx-xxx-xxxx" // en caso de que la conversión
       este en caché tomar esa y no volver a generarla, validación para senior
       "fullName": "Juan Perez",
       "version": "TUCSON TL",
       "model": "TUCSON",
       "cryptocurrency": "BTC"
 }
}
```

Salida

```
{
 "data": {
       "fullName": "Juan Perez",
       "version": "TUCSON TL",
       "model": "TUCSON",
       "cryptocurrency": "BTC",
       "priceUsd": 25000.0,
       "priceCryptocurrency": 1,
       "date": "2023-06-02:12:01:45",
       "purchaseId": "yyy-yyy-yyy-yyy"
 }
}
```

R3 **API de Reporte:** Generará un reporte en formato JSON que mostrará el total de ventas diarias de un modelo o versión específica de vehículo, en dólares y en la criptomoneda seleccionada.

Ejemplo API Reporte

Entrada

```
{
 "data": {
       "date": "2023-06-02",
       "model": "TUCSON",
       "cryptocurrency": "BTC"
 }
}
```

Salida

```
{
"data": [{
       "date": "2023-06-02",
       "model": "TUCSON",
       "cryptocurrency": "BTC",
       "usdAmount": 245000,
       "cryptocurrencyAmount": 5
 }
}]
```

Requerimientos Técnicos

R4 **Caché de Información:** Implementar un sistema de caché para los datos de precio en tiempo real de las criptomonedas y la lista de versiones de un modelo específico de vehículo.

R5 **Almacenamiento Persistente:** Las compras realizadas se deben almacenar de forma persistente en una base de datos PostgreSQL.

R6 **Almacenamiento Persistente:** Las compras realizadas se deben almacenar de forma persistente en una base de datos PostgreSQL, a ser implementada desde un archivo docker compose.

R7 **Manejo de Errores:** Implementar un manejo de errores, que devuelva un código de estado HTTP apropiado y un mensaje de error si algo sale mal.

R8 **Entradas y Salidas:** Cada API tendrá entradas y salidas definidas, las cuales deben cumplirse en su totalidad

Requerimientos no funcionales

R9 Arquitectura de Software: Diseñar la arquitectura del software teniendo en cuenta la modularidad, escalabilidad y los principios adecuados de arquitectura y diseño de software.

R10 Implementar pruebas unitarias

R11 Implementar pruebas de integración.

R12 Contenerización de los microservicios.