CryptoCurrency Wallet Simulator API

Introduction

Using API services from CryptoCompare, we need to generate an API that will allow to simulate currency wallets allowing the operations over it.

Description

The simulator design will be a simple POC, it will allow CRUD operations of the Wallets, and allow for the transfer operations in, and between wallets.
The prices and available symbols have to be obtained from the APIs exposed in CryptoCompare Services (e.g.: Multiple Symbols Service), and have to be obtained when doing the exchanges of the currency.
Bear in mind that the described input and output result content, are not limited to the specified elements, but the specified values should be present.
There is no security implemented for the API, so all the operations are unrestrained to simplify the implementation.

Get All CryptoCurrencies Service

Return all the crypto currencies that are available with prices.

Returns: All the available currencies with prices.

Create Wallet Service

Create the wallet in the simulator.

Input: This should include a friendly name for the wallet, and the values stored initially, if any.

Returns: The created wallet should be created with an id generated in the application.

Get Wallet Service

Get a Wallet in the simulator.

Returns: The full wallet information

Update Wallet Service

Update a Wallet in the simulator.

Returns: The updated wallet.

Remove Wallet Service

Removes a Wallet from the simulator

Returns: The removed wallet

Buy Currency Service

Buys a specified currency using the specified amount of a specified currency. The conversion should be obtained from the real values (proposed API).

Input: This should include the amount and currency, and the destination currency and wallet

Returns: The information of the wallet, including currency changes if any.

Transfer Values for two Wallets Service

Transfer the amount of a currency from a Wallet to specified currency on another Wallet in the simulator.

Input: This should include the currency and the amount to transfer, as well as the currency and destination to transfer to.

Problem restrictions

- The code should be in Java and related frameworks..

Expected results

The problems should be delivered with its source code (e.g.: github repo.), provide the API documentation, as well as the endpoint of a live sandbox to test it. There is no need for persistence to be implemented.

Nice to haves

- We probably want to cache if possible.
- We need to implement pagination for relevant services

Feel free to ask any questions.