Problem Statement: develop a service, that constantly checks the currency exchange rate from Bitcoin to US-Dollar (1 Bitcoin = x USD).

Rest API Design

1. API to fetch latest price

|  |  |
| --- | --- |
| URL | /last-price/{fromCurrency}/{toCurrency} |
| Http Method | GET |

Request Parameters

|  |  |  |
| --- | --- | --- |
| Type | Description | Mandatory |
| String | The Currency code of the currency that will be converted to | Yes |
| String | The Currency code of the currency to be converted to | Yes |

Example:

GET http://localhost:8080/v1/last-price/BTC/USD

Response

200 OK

{

“from-currency”: “BTC”,

“to-currency”: “USD”

“exchange-rate”:9000,

“timestamp” : “1560694884511”

}

1. Get API to fetch historical data between startDate and endDate

|  |  |
| --- | --- |
| URL | /historical-data?startDate={startDate}& endDate={endDate} |
| Http Method | GET |

Request Parameters

|  |  |  |
| --- | --- | --- |
| Type | Description | Mandatory |
| String | Start date in yyyy-MM-dd format | Yes |
| String | End date in yyyy-MM-dd format | Yes |

Example:

GET http://localhost:8080/v1/ historical-data?startDate={fromCurrency}& endDate={toCurrency}

Response

200 OK

{

"from-currency": "BTC",

"to-currency": "USD",

"daily-exchange-price": {

"2019-06-01": "8553.1267",

"2019-06-02": "8737.3617",

"2019-06-03": "8105.13",

"2019-06-04": "7677.3983",

"2019-06-05": "7785.985",

"2019-06-06": "7806.715",

"2019-06-07": "8001.2767",

"2019-06-08": "7928.5417",

"2019-06-09": "7634.6483",

"2019-06-10": "8016.6917",

"2019-06-11": "7913.8933",

"2019-06-12": "8173.3567",

"2019-06-13": "8232.7533",

"2019-06-14": "8693.765",

"2019-06-15": "8857.0783"

}

}

Assumptions:

1. For historical data the avg-price will be returned along with the max and min values
2. The Exchange Server doesn’t hold data.

Reason:

* 1. The latest-price needs to fetched from the external server.
  2. The Historical data need can have higher SLA as it is most probably used for some sort of analysis.

1. Based on the requirement the Historical-data can be merged with the latest-price api or can be separated out to another microservice to meet SLA requirements.
2. Exchange service runs behind a load Balancers.
3. Https will not be considered in the design
4. A better response to historical data would have been

{

"from-currency": "BTC",

"to-currency": "USD",

"dates": {

"2019-06-10": {

"avg-price": 9000,

"max-price": 9290,

"min-price": 8890

},

"2019-06-11": {

"avg-price": 9010,

"max-price": 9300,

"min-price": 8730

},

"2019-06-12": {

"avg-price": 8990,

"max-price": 8800,

"min-price": 8730

}

}

}

1. With the current design we can enhance the code to work with different currencies.
2. I would prefer directly calling the External Exchange Server rather call another service that will continuously check the Exchange Server for conversion rate, as described in the problem statement. Have a http cache with low TTL (10seconds, assuming the exchange rates don’t change for 10 seconds)
3. If we have to call the Exchange Server for every check period, then we need to store the data within our system and we could compute the daily average, max rate, min rate for day. However, as that will require to store data every x seconds, I am calling the service to get the historical data.
4. For latest price, there is a scheduled thread (runs in a different thread) that will call exchange server for rates start of every minute (configurable).

High level Design

Load balancer

Exchange Service

Exchange Service

BitCoin Server

Exchange Service

Low level design

1. Scheduler Flow:
   1. Scheduler runs every minute
   2. Fetches the exchange rate from the exchange server
   3. Updates the cache

Cache

BitCoin Server

Scheduler Task

Update cache

1. Latest Price Call Flow
   1. Get the value from cache, as the cache always has the updated value
2. Historical Data Flow
   1. Instead of storing all the data in our system, we get it from external system

service

BitCoin Server

Storing data every minute and computing the avg exchange rate for a day would be difficult to test, hence I am calling an external service to get the result.

Configurable parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Description | Comments |
| **config.exchange.server.latest-price.url** | Url to external server to get the latest price | Better to accept only host and compute the url in the code for every type of currency. |
| **config.exchange.server.historical-data.url** | Url to external server to get the historical data | Better to accept only host and compute the url in the code for every type of currency. |
| **config.exchange.check.period.cron** | Cron pattern for running the scheduler | To run every minute use **0 \* \* \* \* ?** |

Improvements:

1. Error Handling
   1. Handle input validations error in a validation framework instead of annotations so that appropriate message can be sent to caller