**Currency Conversion**

* **APIs**

**API for client to retrieve an exchange rate.**

GET

http://localhost:8080/public/conversion/[sourceCurrency]/[targetCurrency]?amount=50.00&startDate=2023-12-28

http://localhost:8080/public/conversion/USD/EUR?amount=50.00&startDate=2023-12-28

{

    "conversion": [

        {

            "startDate": "2023-12-28 19:56:00",

            "sourceCurrency": "USD",

            "targetCurrency": "EUR",

            "amount": 50,

            "value": 45.0245

        }

    ]

}

**API to obtain exchange rate from feed or direct customer data load.**

GET

http://localhost:8080/public/exchangerate

{

    "exchangerate": [

        {

            "dateRate": "2023-12-27 19:56:00",

            "currencysymbol": "USD",

            "valueRate": {

"KRW": 1291.29411,

"COP": 3834.64364,

"EUR": 0.90049

}

        },

        {

            "dateRate": "2023-12-28 10:15:00",

            "currencysymbol": "USD",

            "valueRate": {

"KRW": 1291.29411,

"COP": null,

"EUR": 0.90049

}

        },

        {

            "dateRate": "2023-12-29 21:06:00",

            "currencysymbol": "USD",

            "valueRate": {

"KRW": 1291.29411,

"COP": 0,

"EUR": 0.90049

}

        },

        {

            "dateRate": "2023-12-29 21:06:00",

            "currencysymbol": "USD",

            "valueRate": {

"KRW": 1291.29411,

"COP": "-",

"EUR": 0.90049

}

        }

    ]

}

**Path parameters**

|  |  |
| --- | --- |
| **Name** | **API** |
| 1. Source Currency | sourceCurrency |
| 1. Target Currency | targetCurrency |
| 1. Request Start Date | startDate |
| 1. Effective Start Date | effectiveDate |
| 1. Exchange Rate | exchangeRate |
| 1. Currency Symbol | currencySymbol |
| 1. Currency Name | currencyName |
| 1. Date Rate | dateRate |
| 1. Amount | amount |
| 1. Value Rate | valueRate |
| 1. Converted Value | value |

* **Test Cases**

Assignment was resolved using Postman.

1. **Test Case**

Verify the API has a response status ok.

***Test Case Based on Requirements****:*

*We need to provide an API for client applications or users to retrieve an exchange rate for a source currency, a target currency, and a specific date.*

* 1. **Test Case Description**

Test the API returns a status 200.

pm.**test**("Status code is 200", **function** () {

    pm.response.to.have.status(200);

});

1. **Test Case**

Verify that the valueRate field always returns a value.

***Test Case Based on Business Rule***:

*If an exchange rate is missing for a date, then the currency exchange resolution logic should go back in time trying to find the exchange rate with the closest effective start date.*

* 1. **Test Case Description**

Test the valueRate field does not contain null, zero, hyphens, empty or undefined values. For example: the valueRate “COP”

pm.**test**("Validate the exchange rate field does not contain null, zero, hyphens or empty value", **function** () {

    var jsonData **=** pm.response.json();

    const valueRate **=** Object.**keys**(jsonData.exchangerate.valueRate).length;

**for** (var i **=** 0; i **<** valueRate.length; i**++**){

  pm.expect(jsonData.exchangerate[0].valueRate[i].COP).is.not.null;

      pm.expect(jsonData.exchangerate[0].valueRate[i].COP).is.not.equals(0);

      pm.expect(jsonData.exchangerate[0].valueRate[i].COP).is.not.string;

      pm.expect(jsonData.exchangerate[0].valueRate[i].COP).is.not.empty;

      pm.expect(jsonData.exchangerate[0].valueRate[i].COP).is.not.undefined;

    }

});

1. **Test Case**

Verify the API contains source and target currency fields.

***Test Case Based on Business Rule:***

*If no exchange rates exist with the source and target currency pair, then the currency exchange resolution logic should look for a triangular conversion.*

* 1. **Test Case Description**

Test the API contains the fields sourceCurrency and targetCurrency.

pm.**test**("Validate the API contains the fields sourceCurrency and targetCurrency", **function** () {

    var jsonData **=** pm.response.json();

    pm.expect(pm.response.text()).to.include("sourceCurrency");

    pm.expect(pm.response.text()).to.include("targetCurrency");

    pm.expect(jsonData.conversion[0].sourceCurrency).to.eql("USD");

    pm.expect(jsonData.conversion[0].targetCurrency).to.eql("EUR");

    pm.expect(jsonData.conversion[0].sourceCurrency).to.be.a('string');

    pm.expect(jsonData.conversion[0].targetCurrency).to.be.a('string');

    pm.expect(jsonData.conversion[0].sourceCurrency).to.have.lengthOf(3);

    pm.expect(jsonData.conversion[0].targetCurrency).to.have.lengthOf(3);

});

1. **Test Case**

Verify the API response time.

***Test Case Based on Requirements****:*

*We need to provide an API for client applications or users to retrieve an exchange rate for a source currency, a target currency, and a specific date.*

* 1. **Test Case Description**

Test the API response time is less than 1000 ms.

pm.**test**("Response time is less than 1000ms", **function** () {

    pm.expect(pm.response.responseTime).to.be.below(1000);

});

1. **Test Case**

Verify that the dateRate field always returns a information.

***Test Case Based on Business Rule:***

*If an exchange rate is missing for a date, then the currency exchange resolution logic should go back in time trying to find the exchange rate with the closest effective start date.*

**5.1 Test Case Description**

Test the dateRate field does not contain null, zero, hyphens, empty or undefined values.

pm.**test**("Validate the dateRate field does not contain null, zero, hyphens, empty or undefined values", **function** () {

    var jsonData **=** pm.response.json();

    const valueRate **=** Object.**keys**(jsonData.exchangerate.valueRate).length;

**for** (var i **=** 0; i **<** valueRate.length; i**++**){

      pm.expect(jsonData.exchangerate[0].dateRate[i]).is.not.null;

      pm.expect(jsonData.exchangerate[0].dateRate[i]).is.not.equals(0);

      pm.expect(jsonData.exchangerate[0].dateRate[i]).is.not.string;

      pm.expect(jsonData.exchangerate[0].dateRate[i]).is.not.empty;

      pm.expect(jsonData.exchangerate[0].dateRate[i]).is.not.undefined;

    }

});

1. **Test Case**

Verify that the dateRate field has a valid date format.

***Test Case Based on Business Rule:***

*If an exchange rate is missing for a date, then the currency exchange resolution logic should go back in time trying to find the exchange rate with the closest effective start date.*

**6.1 Test Case Description**

Test the dateRate field has a date format.

pm.**test**("Validate dataRate format", **function** () {

    var jsonData **=** pm.response.json();

    let dateRate **=** jsonData.exchangerate[0].dateRate;

    let moment **=** **require** ('moment');

    pm.expect(dateRate).to.include(moment(dateRate).format('YYYY-MM-DD HH:mm:ss'));

});