RESTful Notes API Guide

Depanker Sharma

Table of Contents

Overview

These are the API's for importing financial data and outputing the report accordingly.

Chapter 1. General

1.1. HTTP verbs

Verb	Usage
`GET`	Used to retrieve a resource (idempotent)
`POST`	Used to create a new resource
`PUT`	Used to update an existing resource (idempotent)
`DELETE`	Used to delete an existing resource

Chapter 2. HTTP status codes

RESTful notes tries to adhere as closely as possible to standard HTTP and REST conventions in its use of HTTP status codes.

Status code	Usage
`200 OK`	The request completed successfully
`202 Accepted`	The request has been accepted for processing, but the processing has not been completed. The request might or might not be eventually acted upon, and may be disallowed when processing occurs.
`204 No Content`	An update to an existing resource has been applied successfully
`400 Bad Request`	The request was malformed. The response body will include an error providing further information
`404 Not Found`	The requested resource did not exist
`409 Not Found`	Attempt to create a duplicate resource
`500 Internal server error`	Any Unhandled

Chapter 3. Errors

Whenever an error response (status code >= 400) is returned, the body will contain a JSON object that describes the problem. The error object has the following structure:

```
HTTP/1.1 400 Bad Request
Content-Type: application/json;charset=UTF-8
Content-Length: 62

[ {
  "field" : "fieldName",
  "message" : "Error message"
} ]
```

Resources

Chapter 4. Register Company

id and version fields should be removed from all requests. Mapping fields is the information about csv headers when parsing product csv

Request

```
$ curl 'http://localhost:8080/register-company' -i -X POST \
    -H 'Content-Type: application/json' \
    -d '{
  "id" : null,
  "version" : null,
  "companyName" : {
    "companyName" : "A"
  },
  "fieldMapping" : {
    "productFieldName" : "Description",
    "costFieldName" : "Purchase price",
    "currencyFieldName" : "Currency",
    "productCompanyIdFieldName" : "Prod. Id"
  },
  "products" : null,
  "orders" : null
}'
```

Response

```
HTTP/1.1 200 OK
```

4.3. Error

Request

```
$ curl 'http://localhost:8080/register-company' -i -X POST \
    -H 'Content-Type: application/json' \
    -d '{
  "id" : null,
  "version" : null,
  "companyName" : {
    "companyName" : null
  },
  "fieldMapping" : {
    "productFieldName" : null,
    "costFieldName" : null,
    "currencyFieldName" : null,
    "productCompanyIdFieldName" : null
  },
  "products" : null,
  "orders" : null
}'
```

Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json
Content-Length: 423

[ {
  "field" : "fieldMapping.currencyFieldName",
  "message" : "must not be blank"
}, {
  "field" : "companyName.companyName",
  "message" : "must not be blank"
}, {
  "field" : "fieldMapping.costFieldName",
  "message" : "must not be blank"
}, {
  "field" : "fieldMapping.productCompanyIdFieldName",
  "message" : "must not be blank"
}, {
  "field" : "fieldMapping.productFieldName",
  "message" : "must not be blank"
} ]
```

Chapter 5. Upload products

Request

```
$ curl 'http://localhost:8080/upload-product' -i -X POST \
    -H 'Content-Type: multipart/form-data' \
    -F 'productFile=@A.csv;type=text/csv'
```

Response

```
HTTP/1.1 200 OK
```

5.3. Error

Request where company is not register

```
$ curl 'http://localhost:8080/upload-product' -i -X POST \
    -H 'Content-Type: multipart/form-data' \
    -F 'productFile=@A.csv;type=text/csv'
```

Response

```
HTTP/1.1 403 Forbidden
```

Chapter 6. Upload sales report

Request

```
$ curl 'http://localhost:8080/upload-order' -i -X POST \
    -H 'Content-Type: multipart/form-data' \
    -F 'salesFile=@Sales.csv;type=text/csv'
```

Response

```
HTTP/1.1 200 OK
```

6.3. Error

Request where company is not register

```
$ curl 'http://localhost:8080/upload-order' -i -X POST \
    -H 'Content-Type: multipart/form-data' \
    -F 'someFile=@Sales.csv;type=text/csv'
```

Response

```
HTTP/1.1 400 Bad Request
Content-Type: application/json
Content-Length: 97

[ {
  "field" : "salesFile",
  "message" : "Required request part 'salesFile' is not present"
} ]
```

Chapter 7. Sales report

Request

```
$ curl 'http://localhost:8080/sales-report/A' -i -X GET
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 205

{
  "response" : [ {
    "orderNumber" : 1,
    "orderDate" : "01/27/20",
    "total" : 101.0,
    "lineItems" : [ {
      "product" : "P1",
      "quantity" : 1,
      "salesPrice" : 101.0
    } ]
  } ]
}
```

7.3. Error

Request when not data is present

```
$ curl 'http://localhost:8080/sales-report/A' -i -X GET
```

Response

```
HTTP/1.1 404 Not Found
```

Chapter 8. Revenue report

Request

```
$ curl 'http://localhost:8080/revenue-report/A' -i -X GET
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 148

{
  "response" : [ {
    "year" : 2019,
    "total" : 99.9,
    "revenueDTOS" : [ {
      "month" : "2019-1",
      "revenue" : 99.9
    } ]
  } ]
}
```

8.3. Error

Request where company is not register

```
$ curl 'http://localhost:8080/revenue-report/A' -i -X GET
```

Response

```
HTTP/1.1 404 Not Found
```

Request when no sales data is present

```
$ curl 'http://localhost:8080/revenue-report/A' -i -X GET
```

Response

```
HTTP/1.1 404 Not Found
```

Chapter 9. Net-Profit report

Request

```
$ curl 'http://localhost:8080/net-profit-report/A' -i -X GET
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 152

{
  "response" : [ {
    "year" : 2019,
    "netProfit" : 99.9,
    "revenueDTOS" : [ {
      "month" : "2019-1",
      "revenue" : 99.9
    } ]
  } ]
}
```

9.3. Error

Request when company is not registered

```
$ curl 'http://localhost:8080/net-profit-report/A' -i -X GET
```

Response

```
HTTP/1.1 404 Not Found
```

Request when not data is present

```
$ curl 'http://localhost:8080/net-profit-report/A' -i -X GET
```

Response

```
HTTP/1.1 404 Not Found
```