Products API - Response Status Codes

Context:

The CDS Specifications provide a reasonable idea of the HTTP Status codes that should be returned by each of the APIs. However there are grey areas where we as team need to make a decision. The table below shows the behaviour of the big 4 banks. It also includes a column of the recommendation for the BEN Common Solution.

#	Scenario	CBA	Westpac	NAB	ANZ	Bendigo's preferred Solution
1	Brand Name Invalid	400	400	200 (0 Records)	200 (0 Records)	400
2	Product Category is Invalid	400	400	400	400	400
3	Page out of range (e.g Valid pages are 5 and you enter 10)	200 (0 Records)	422	200 (0 Records)	200 (0 Records)	422
4	Page > 1000	200 (0 Records)	422	200 (0 Records)	200 (0 Records)	422
5	Page Number/Page size not a positive Integer	400	400	400	400	400
6	x-min-v not an Positive Integer	Could not get any response	Could not get any response	406	400	400
7	Non existent Product ID (in get Product Details)	200	404	400	403	404
8	Trailing / in a GET Products request	200 (same as GET Products without a trailing slash)	200 (same as GET Products without a trailing slash)	404	200 (same as GET Products without a trailing slash)	200 (Ignore the trailing slash)

#	Scenario	CBA	Westpac	NAB	ANZ	Bendigo's preferred Solution
9	No matching products when filters/parameters are specified	200 (0 Records)	200 (0 Records)	200 (0 Records)	200 (0 Records)	200 (0 Records)
10	x-v is a positive Integer but is not a supported version	406	406	406	406	406
11	Product ID is an Invalid ASCII String	200	400	400	403	400
12	Valid version between x-min-v and x-v however x-v is not an implemented version					200

Table 1: Scenarios and expected HTTP Status Codes

Scenario #1 : Brand name is Invalid

The product APIs have Brand name as a parameter. If the brand name is invalid (ie. not a brand of the bank) what is the expected behaviour.

The brand names (or the codes/brand IDs that represent the various brands) might not be known to the callers. The brand IDs that will be used in the solution are documented here.

Recommendation: Return a 400 Error with a meaningful error code and error message.

A sample error message is shown below

```
{
    "errors": [
        {
            "code": 400,
            "title": "Bad Request",
            "detail": "TEST is not a valid brand. Valid brands are 'Bendigo','Adelaide','Rural', 'Delphi'"
        }
    ]
}
```

Rationale : Returning a 200 OK with zero results does not provide any clue to the Clients. Returning the list of accepted values in the error message provides an opportunity for the clients to make a subsequent successful call as they now have the list of brands that are accepted.

This request is considered as a *malformed* request.

Scenario #2 : Product Category is Invalid

The CDS specifications specify the list of valid values for Product Category. However it does not prevent a client from specifying a value outside this list of valid values (e.g TEST).

Recommendation: Return a 400 Error with a meaningful error code and error message.

A sample error message is shown below

```
{
    "errors": [
        {
            "code": "ERR-PB-04",
            "title": "product-category is invalid",
            "detail": "product-category must be a valid enumeration."
        }
    ]
}
```

Rationale : The Clients have all the information to create a valid request. This request is considered as a *malformed* request.

Scenario #3 : Page Size out of range

The Page size returned by products is dependent on the number of products and the parameters specified in the request. The Users are expected to look at the first page results and get the list of pages from the meta section. If the user chooses to specify a value that is beyond this range, what is the expected behaviour?

Recommendation: Return a 422 Error with a meaningful error code and error message.

A sample error message is shown below

```
{
    "errors": [
        {
            "code": ERR-PB-422,
            "title": "Invalid Request",
            "detail": "Page requested is out of page range"
        }
    ]

}
```

Rationale : The User have ignored the information from the results page and have entered an invalid value. The choices are a 400 for a malformed request or a 422 which is used for Scenario#4 below. Using a consistent error code 422 for Invalid page size (whether out of range or greater than 1000)is appropriate.

Scenario #4 : Page Size greater than 1000

The CDS specifications is clear in this area, however the Big 4 have a differing behaviour.

Recommendation: Stick with the Specification and return a 422 Error with a meaningful error code and error message.

A sample error message is shown below

```
{
    "errors": [
        {
            "code": ERR-PB-422,
            "title": "Invalid Request",
            "detail": "Page requested is out of page range"
        }
    ]

}
```

\

Rationale : The Specifications are clear and there is no need to deviate from the specifications. .

Scenario #5 : Page Number/Page-size not an Integer Value

The CDS specifications state that both Page number and page size should be a Positive integer. Any value other than this is considered a malformed input.

Recommendation: Stick with the Specification and return a 400 Error with a meaningful error code and error message.

A sample error message is shown below

```
{
    "errors": [
        {
            "code": ERR-PB-422,
            "title": "Invalid page parameter: not an integer",
            "detail": "Page only accepts positive integer (zero excluded)"
        }
    ]
}
```

Rationale : The Specifications are clear and there is no need to deviate from the specifications. .

Scenario #6 : x-min-v is not a Positive Integer

x-min-v is defined as a positive Integer. Any value that is not a positive integer should return an error result.

Recommendation: Return a 400 Error with a meaningful error code and error message.

A sample error message is shown below

```
{
    "errors": [
        {
            "code": ERR-PB-422,
            "title": "Unacceptable header value",
            "detail": "Version number x-min-v:-5 is invalid: must be a positive
integer"
        }
    ]
}
```

Rationale : The specification specifies the valid values for the fields. If an invalid value is specified then it should be treated as Malformed User request, hence return a 400.

Scenario #7 : GET request for a product ID that does not exist

Typically, a 404 is returned for a resource that does not exist. However the CDS specifications in its list of HTTP status codes does not list 404 at all.

Recommendation: Return a 404 which is the industry standard for a resource not found..

A sample error message is shown below

```
{
    "errors": [
        {
            "code": ERR-PB-422,
            "title": "Not Found",
            "detail": "Product not found"
        }
    ]
}
```

Rationale : Though the specifications does not state 404 as a valid HTTP status code we will use this code as this is the industry standard status code for resource not found errors.

Scenario #8 : Use a Trailing Slash in a GET Products request

Should this request be treated as a GET products request (ignoring the trailing slash) or should this be treated as a GET Product details with a *null* value for product ID (i.e treat this as a non-existent Product ID similar to Scenario #4)

Recommendation: Be Defensive and ignore the trailing slash and return the response for GET Products API

Rationale : The REST API is not specific in this area. The Big 4 except for one appears to be forgiving/defensive. Let’s give the caller a benefit of doubt and treat is as a GET Products API request.

Scenario #9 : Filters result in no matching records

if the request parameters result in on matching records what is the expected result. There is no ambiguity here. The specification expects the APIs to return a 200 with an empty list.

Recommendation: Follow the specifications and return a 200 status Code.

A sample response message is shown below

```
{
    "data": {
        "products": []
    },
    "links": {
        "self": "https://api.bendigo.com/cds-au/v1/banking/products/?product-category=LEASES"
    },
    "meta": {
        "totalPages": 0,
        "totalRecords": 0
    }

}
```

Rationale : Obey the Specification as it is clear and unambiguous.

Scenario #10 : x-v is a Positive Integer but specified Version not Implemented

x-v is defined as a positive Integer. Users can specify a x-v that has not been implemented. E.g User enter x-v as “5”.

Recommendation: Return a 406 Error with a meaningful error code and error message.

A sample error message is shown below

```
{
    "errors": [
        {
            "code": ERR-PB-406,
            "title": "Unable to retrieve list of products",
            "detail": "Unable to select an endpoint version. Implemented: 1"
        }
    ]

}
```

Rationale :

Scenario #11 : Product ID is an Invalid ASCII String

Product ID is defined as an ASCII String. If Users specify an invalid ASCII String e.g “TEST!#@ “

Recommendation: Return a 400 Error with a meaningful error code and error message.

A sample error message is shown below

```
{
    "status": 400,
    "errors": [
        {
            "code": "ERR-PB-400",
            "title": "Bad request",
            "detail": "Product ID is an Invalid ASCII String"
        }
    ]
}
```

Rationale : Product ID is defined as an Ascii String. Users specifying an invalid value is considered a Bad request (i.e 400 Status Code). This is consistent with the way we have handled bad inputs from the consumers.

Scenario #12 : Valid version between x-min-v and x-v however x-v is not an implemented version

Note: The big 4 have only implemented V1 of the product APIs. So we are not able to meaningfully make observations on their response behaviour.

Users can specify any positive integer for x-v (say 100). This version will likely be not implemented by the Data Holder? What is the expected behaviour?

Assumption: Both x-min-v and x-v are positive integers .

An example of this scenario could be:

- x-min-v =**1**
- x-v = **5**
- Implemented versions : 1 and 2.

Recommendation: Return a **200** with the highest supported version of the Implementation . In the example above the results of v2 should be returned. Th response header x-v should carry the value of **2**.

A sample response message is shown below


```
{
    "data": {
        "products": [
{
                "productId": "A6B5-CC43-985D-EFF8",
                "effectiveFrom": "2020-01-05T00:00:00.000Z",
                "effectiveTo": "2050-01-30T00:00:00.000Z",
                .....
]
    },
    "links": {
        "self": "https://api.bendigo.com/cds-au/v1/banking/products/?product-category=LEASES"
    },
    "meta": {
        "totalPages": 7,
        "totalRecords": 1
    }

}
```


Rationale : Obey the specifications!

The Specifications state the following about x-v and x-min-v.

“The holder should respond with the highest supported version between **x-min-v** and **x-v**.”

If there is an implementation version(s) between these two values, then the Data Holder is expected to return the highest supported implementation version.