Data Standards Body
Technical Working Group

Decision 008 – Use Of Pluralisation

Contact: James Bligh

Publish Date: 15th September 2018

Decision Approved By Chairman: 26th September 2018

Context

The standard used for entity collections in the UK standards aligns with the recommendations of JSONAPI.org. This recommendation is a generally accepted model for the definition RESTful APIs.

Under these standards a collection is accessed using a URI representing the plural of the entity and a single record is accessed at the same URI by supplying the an ID. For example:
GET …\accounts - Returns an array of accounts
GET …\accounts\{id} - Returns the detail of a specific account

This approach is simple and consistent but it does introduce issues for an API regime that represents complex entity relationships.

Some of the specific the issues that will definitely be applicable in the API standards based on current scope expectations are as follows:

Issue 1: Collection attributes cannot be easily represented
Normally an attribute of an entity is specified at the next level of the URI. For example the balance of an account can be specified as:
GET …\accounts\{id}\balance - Returns the balance of a specific account

Using the JSONAPI.org recommendation an attribute on a collection becomes difficult to represent. For example, if there was a need to return the net position for a customer across all accounts the ideal way to represent this would be:
GET …\accounts\netposition - Returns the net position across all accounts for a customer

Issue 2: Bulk retrieval
If there is a need to get record level information across multiple records there is no clear way to do this. With the UK standards bulk retrieval is handled as follows:
GET …\accounts\{id}\transactions - Returns transactions for a specific account
GET …\transactions - Returns transactions for multiple accounts

This model results in a confusing series of root level entities that are actually variations on an existing collection. In same cases this may introduce new collisions if record level data applies to two different collections of different entities.

Issue 3: Complex filtering using a POST

If I have a collection under the JSONAPI.org recommendation it is likely to be represented as follows:

GET …\accounts - Returns multiple accounts

To filter the list of accounts returned the client would apply filters using query parameters. If a complex filter was required then the use of query parameters may be inadequate to represent the complexity of the filter. In banking this is a common use case for business customers that would like to produce reports across multiple accounts out of a portfolio of potentially hundreds.

The preferred approach to this would be to supply a complex filter in JSON using a POST call. This would look something like:

POST …\accounts - Returns multiple accounts using a complex filter

Unfortunately a POST to a collection, under the model described, should actually result in the creation of a new record in the collection. This is a collision. The same URI cannot be used for record creation and complex filtering. To resolve this a search resource is needed that is non-obviously related to the collection.

Decision To Be Made

Determine the approach to handling collections in the API standards.

Feedback Provided

The original proposal and the associated feedback can be found at:
https://github.com/ConsumerDataStandardsAustralia/open-banking/issues/8

Feedback was fairly consistent in support of option 1 of the original proposal. The feedback also acknowledged that there were valid circumstances where collections would need sub-resources and complex queries using POST. This decision is therefore an attempt at a satisfactory consensus on these two positions.

Decision For Approval

The decision regarding pluralisation is to follow the UK model (and JSONAPI.org recommendation). This was listed as option 1 in the original proposal document.

Under this model collections, individual members and collection sub-resources would be accessed as follows:

GET …\accounts	- Returns an array of accounts
GET …\accounts\{id}	- Returns the detail of a specific account
GET …\accounts\transactions	- Returns the transactions of multiple accounts
GET …\accounts\{id}\transactions	- Returns the transactions of a specific account
POST …\accounts	- Create a new account
POST …\accounts\search	- Returns an array of accounts based on a complex query

Note that these are examples to show usage and are not pre-emptive of the actual account URIs to be defined in the API standards.

This decision has the following implications that should be noted:

- We have no clear pattern for complex search queries that require a POST except for the creation of a dedicated sub-resource for the query. This is because a POST to a collection is reserved for the creation of a new collection member. The sub-resource will, in effect, introduce verbs into the URI structure since a query is an action, not a resource (ie. the "search" URI listed in the example above).
- To allow for sub-paths of resources we will need to allow static and dynamic path elements at the same level, as described in feedback entries above. It is assumed that this will not cause technical problems for providers and that their technology stacks are able to accommodate prioritisation of routes and the application of policies.