**STS Endpoint Summary**

**2020-Apr-09**

1. In general, and endpoint is provided for each “item” in the bento-metamodel, except for “relationship”.
2. General endpoints that return “collections” of items. For example, /value\_sets will return a collection of “value\_set” items – more specifically, an array of “value\_set” elements which each “value\_set” element being composed of the “id” and url”. These individual elements will **not** describe links/relations to other items; e.g. each “value\_set” element will NOT list the terms that belong to that value set. To find the terms that belong to that value set, the specific value set endpoint should be accessed (e.g. /value\_sets/:value\_set\_id).
3. For cases for there are multiple property objects with the same handle (e.g. file\_type), there are two approaches: (a) leave objects separate (my preference), and (b) to merge objects. However, how the merge the objects when there are differences can be complicated, as shown below. I have examined approaches to merge based on shared concepts, but I have not figured it out yet. I have not included other objects such as concepts, but it makes the merging even more complicated

|  |  |
| --- | --- |
| Treat each (:property {handle:”file\_type”}) as unique, and list two objects | Merge the two objects into a single object |
| {  "handle": "file\_type",  "model": "ICDC",  "value\_domain": "enum",  "is\_required": "1"  }  {  "handle": "file\_type",  "model": "CTDC",  "value\_domain": "enum",  "is\_required": "1"  } | {  "handle": "file\_type",  "model": ["ICDC","CTDC ]  "value\_domain": ["enum", "enum"]  "is\_required": ["1", “1”]  } |
| {  "handle": "uuid",  "model": "ICDC",  "value\_domain": "None",  }  {  "handle": "uuid",  "model": "CTDC",  "value\_domain": "string",  "is\_required": "1"  } | ?? |

|  |  |  |  |
| --- | --- | --- | --- |
| Endpoint | Response | Notes | Useful parameters |
| /nodes | Will return list of all nodes (will not show properties of said nodes) |  | ?model=  where model = node.model |
| /nodes/:node\_handle | Show (node\_handle:node)  + (:property)  + (:concept)  w/ relationships to other nodes |  | where model = node.model |
| /properties | Will return list of all properties (incl handle, value\_domain, model, is\_required) (will not show value sets of said models) |  | ?model=  where model = property.model |
| /properties/:property\_handle | show (property\_handle:property)  + (:value\_set)  + (:concept) |  | ?model=  where model = property.model |
| /value\_sets | Lists all value sets |  | ?model=  where model = property.model |
| /value\_sets/:value\_set\_id | Show (value\_set\_id:value\_set)  + (:term)  + (:origin) | Will also eventually be show how the specific value set can be represented by specific ‘concept’ in external org | ?model=  where model = property.model |
| /value\_sets/:value\_set\_id/terms/ | Lists terms connected to value set |  |  |
| /value\_sets/:value\_set\_id/terms/:term\_id | Shows (term\_id:term) | **This will do term validation** | ?model=  where model = property.model |
| /concepts | Lists all concepts |  |  |
| /concepts/:concept\_id | Show (concept\_id:concept)  + ()-[:has\_concept]-> (concept\_id)  + ()-[:represents]-> (concept\_id) |  |  |
| /concepts/:concept\_id/terms/:term\_id | Shows (term\_id:term) | **This will do term validation** |  |
| /origin | Lists all external origin |  |  |
| /origin/:origin\_name | Lists everything we know from this origin |  |  |
| /origin/:origin\_name/:origin\_concept\_id | Shows the concept for this origin |  |  |