Project Origin Data Repository REST Endpoints:

|  |  |
| --- | --- |
| **Endpoint** | **Description** |
| /wpaper | Calculation Workpaper API   * POST (upload/create) * PUT – pending * GET Search * GET File (binary xlsx file) * GET List (list of xlsx files by client) * DELETE (binary xlsx file and associated metadata) |
| /template | Template API   * POST (upload/create) * GET Search * GET File (binary template file) * GET List (list of template files by client) * DELETE (binary template file and associated metadata) |
| /datarequest | Data Request API   * POST (upload/create) * GET Search (data request JSON docs) * GET Doc (data request JSON doc) * GET List (list of data request JSON docs by client) * DELETE (data request JSON doc)   New APIs   1. GET dname elements by request id and dname (optional) 2. GET list of dnames |
| /suggest | Type Ahead |

**Notes from 5/13/2015**

Changes for /wpaper

* Stay with clientId but it means firmId - Pass Oracle primary Key for client id
* Use workPaperName not the fileName
* POST - must have clientId, workPaperId, version, workPaperName
* Use standard error messaging

Features for /wpaper PUT – update:

* Need ability to change workPaperName
* Use payload to update any element
* Only store most recent binary
* lastmodified user
* Pass in Username and lastModifed User
* Add user info to qstring for now
* Store ONESOURCE user info only.

A simple search portal for both the template and user data is available here. => <http://glm-ml-dev.amers1.cis.trcloud:8010/>

**/template**

| # | Type | Action |
| --- | --- | --- |
| 1 | POST (create) | Uploads a new template file that physically resides on the local client directory.  Notes:   * A template name parameter must be provided by the app server that is unique. * Each version of the template will result in a unique template name. * Folder is created under the template directory in MarkLogic with the name of that template. * Folder contains the binary version of the excel file that is being uploaded as well as the xml version of the file containing pertinent data for the file that will be used for searches. |
| 2 | GET (File) | Returns specific template spreadsheet file in excel (binary) format. |
| 3 | DELETE | Soft Delete – Archives the Template file and its metadata. This has not been fully flushed out yet and may not be needed. |

**/datarequest**

| # | Type | Action |
| --- | --- | --- |
| 1 | POST (create) | Creates a new data request record.  Notes:   * Data Request Identifier (GUID) and Template Name must be provided as parameters. * New folder is created under the /datarequest directory that uses the Template Name parameter. * System will create an XML/JSON file and will store it in the New Template Name directory. * XML/JSON file will have a set of key/value pairs of the defined names for which the template was filled out with alongside any other relevant metadata for that data request. |
| 2 | GET (merged file) | Returns a merged spreadsheet (binary) xlsx file containing the template with the filled out datarequest data.  Template name and data request identifier (GUID) are passed in as parameters. |
| 3 | GET (datarequest xml/json) | Returns the raw xml/json of the datarequest given the template name and data request identifier (GUID) as parameters. |
| 4 | PUT (update existing datarequest) | Updates the existing XML/JSON of a datarequest given the Template Name and Data Request identifier (GUID) as parameters.  Notes:   * Once users fill out the datarequest, they can continuously update it. * Application will do the following:   1. GET the existing datarequest XML/JSON document from MarkLogic   2. Compare existing XML/JSON doc with new data being passed in   3. Create a new XML/JSON document   4. Store revised XML/JSON document in MarkLogic. |

Endpoint Details:

1. Template API – Used for Workpaper Template CRUD Operations

| # | Type | Parameters | Action | |
| --- | --- | --- | --- | --- |
| 1 | POST | **Parameters:**  rs:filename – required  **Examples:**  /template  /template?fs:filename=C590300 | Uploads a new template file that physically resides on the local client directory.  The filename must be provided.  The payload is the binary Excel file.  API returns the template ID and URI.  The binary file with its respective metadata file is stored in the dedicated directory (see image). | |
| **Curl Commands:**  In this case, the template file is indicated by “./C5903000.xlsx”.   1. POST Template File with desired filename.   curl -X POST --data-binary @./C5903000.xlsx \  --header "Content-Type:application/vnd.openxmlformats-officedocument.spreadsheetml.sheet" \  http://glm-ml-dev.amers1.cis.trcloud:8017/v1/resources/template?rs:templateName?rs:templateName=HomeDepot-2014-Q2-Earnings\_11251437686954615217\_16 | | |
| 2 | GET File | **Parameters:**  rs:id – Template ID  **Example:**  /template?rs:id=8216640002390149622 | | Returns specific template spreadsheet file. |
| **Curl Commands:**   1. GET Binary by Template ID 8216640002390149622   curl --anyauth --user grusso:password -o templateC5903000.xlsx -X GET \    -H "Accept: application/vnd.openxmlformats-officedocument.spreadsheetml.sheet" \    -r "0-511999" \    http://glm-ml-dev.amers1.cis.trcloud:8010/v1/resources/template?rs:id=8216640002390149622 | | |
| 3 | DELETE | **Parameters:**  rs: templateName  **Examples:**  /template?rs:templateName = HomeDepot-2014-Q2-Earnings\_11251437686954615217\_16 | Deletes the template directory and its contents. | |
| **Curl Commands:**   1. DELETE Template Directory   curl --anyauth --user grusso:password -X DELETE \  -H "Accept: application/xml" \  http://glm-ml-dev.amers1.cis.trcloud:8017/v1/resources/template?rs:templateName= HomeDepot-2014-Q2-Earnings\_11251437686954615217\_16 | | |

1. Data Request API – Used for Workpaper Data Request CRUD Operations

| # | Type | Parameters | Action / Curl command to Test |
| --- | --- | --- | --- |
| 1 | POST | **Parameters:**  rs:templateid – associates the json name/value pairs with the specific spreadsheet template file.  **Example:**  /datarequest?rs:templateId=17395420970371531535  Input   * URI parameter: template Id * Payload: Name/value pairs | Creates a new User Data file.  The payload is the User Data file.  Returns the User Data File ID and maybe the URI.  JSON Format:  { "dnames": [  {"name":"I\_0001", "value":"1001.25"},   {"name":"I\_0002", "value":"1002.25"},   {"name":"BegMnA.block","value":"44442.98"},  {"name":"EndMnA.block","value":"17772.34"},  {"name":"EndMnA.block","value":"27772.34"} ] } |
| **Curl Commands:**   1. POST User Data   curl --anyauth --user grusso:password -X POST \  -H "Content-Type: application/json" \  -d '{"dnames":[{"name":"I\_0001","value":"1111.25"},{"name":"I\_0002","value":"2222.25"},{"name":"EndMnA.block","value":"27772.34"}]}' \  http://glm-ml-dev.amers1.cis.trcloud:8017/v1/resources/datarequest?rs:id=17395420970371531535   1. POST User Data File – Used above json but is stored in file called ./data1.txt   curl --anyauth --user grusso:password --data @./data1.txt -X POST \  -H "Content-Type: application/json" \  http://glm-ml-dev.amers1.cis.trcloud:8017/v1/resources/datarequest?rs:id=17395420970371531535 | |
| 2 | GET Merged Doc | **Parameters:**  rs:id – User Data ID  rs:merge – true (default value)  **Example:**  /datarequest?rs:id=17395420970371531535 | Returns a merged spreadsheet file.  Returns binary xlsx file. |
| **Curl Command:**  curl --anyauth --user grusso:password -o mergedWorkPaper1.xlsx -X GET \  -H "Accept: application/vnd.openxmlformats-officedocument.spreadsheetml.sheet" \  -r "0-511999" \  http://glm-ml-dev.amers1.cis.trcloud:8017/v1/resources/datarequest?rs:id=17395420970371531535 | |
| 3 |  | **Parameters:**  rs:id – User Data ID  rs:merge – false  **Example:**  /datarequest?rs:id=17395420970371531535&rs:merge=false | Returns the user data document. |
| **Curl Command:**  curl --anyauth --user grusso:password -o mergedWorkPaper1.xlsx -X GET \  -H "Accept: application/vnd.openxmlformats-officedocument.spreadsheetml.sheet" \  -r "0-511999" \  http://glm-ml-dev.amers1.cis.trcloud:8017/v1/resources/datarequest?rs:id=17395420970371531535&rs:merge=false | |