REST API Reference Documentation - 2.6

HINT: you may want to use **double quotes** ("search") to surround your search term if you are having trouble finding the term you need.

On this page:

Resources and Services:

Target audience:

- CAST AI Administrators
- Consumers of information via the REST API

Scope

This REST API is provided

- either for data stored in a **Measurement Database** for a Health Dashboard (**HD**)
- or for data stored in a **Central database** for an Engineering Dashboard (**ED**)

Discovering the REST API

A simple Web Application, called "Simple REST Client, is available to discover the REST API. This Web application allows you to test REST URIs, check JSON response, and use the "preview" form to navigate a resource to linked resources.

WAR archive

If you use the war archive of Dashboards/REST API (eg. com.castsoftware.aip.dashboard.2.0.0-funcrel.war), this web application can be accessed via the following URL:

http://localhost:8080/com.castsoftware.aip.dashboard.2.0.0-funcrel/static/default.html

The "Server" text box in the "Simple REST Client" must contain: **/com.castsoftware.aip.dashboard.2.0.0-funcrel/rest/**

ZIP archive

If you use the zip archive of Dashboards/REST API (eg. com.castsoftware.aip.dashboard.2.0.0-funcrel.zip), this web application can be accessed via the following URL:

http://localhost:8080/static/default.html

The "Server" text box in the "Simple REST Client" must contain: **/rest/**

Uniform Resource Identifiers

All URIs are relative to a domain of a REST Server. Therefore, a REST client must always concatenate an URI with the URI of the REST Server.For example, if the REST Server URL is:

http://localhost:9090/Dashboard-WebService,

and the URI is **AAD/applications**, then the REST server is invoked with the following URL:

http://localhost:9090/Dashboard-WebService/rest/AAD/applications

---

URI Templates

URI templates notation is specified below (see also http://tools.ietf.org/html/rfc6570):

- {string} notation:
  this syntax denotes a string expansion.
- {?parameters} notation:
  this syntax denotes an optional parameter to expand. The URL template is expanded as follow:
  - append "?" to the result string if this is the first defined value or append "&"
  - append the parameter name
  - append "="
  - append a parameter value which is a list of items, with "," as a separator (brackets separators are optional)

Request Patterns

Requests	Content-Type: application/json	Content-Type: text/csv	Comment
GET .../{resources}/{ID}	Yes	No	Report a full representation of a resource
GET ../{resources}?{filters}	Yes	Yes	Report a collection of resources Each item is a short representation of a resource
GET ../{ressources}-summary	Yes	No	Report resources counters
POST ../{resources}	Yes	No	Insert a collection of resources. This action is not idempotent, each call will insert new items
DELETE ../{resources}	Yes	No	Remove a collection of resources or items.
PUT ../{resources}	Yes	Yes	Update a collection of resources. This action is idempotent, a second call will not change anything In CSV mode, replace a collection of resources

POST, PUT, DELETE payloads are always a collection of resources. Note that a resource is an item with an "href" attribute.

Resource Representation

Media Types

This REST API supports the following media-type:

Media Type	Resource Representation
application/json	A JSON text (for all resources)
text/plain	A plain text (for file contents) Note this will only list the first 10 violations.
text/csv	A Comma Separated Value Text. Separator is ";" Decimal Separator is "." Empty column is represented with "null" value.
application/vnd.openxmlformats- officedocument.spreadsheetml.sheet	An Excel document with "xlsx" extension

JSON Representation

A JSON representation is a structure compliant with JSON format.Each property can be of type

- Array (a collection of items)
- String
- Integer (JSON number)
- Double (JSON number)
- Number
- Structure (a nested structure)
- URI (a resource reference).
- Date: JSON structure
  - "time" attribute: date as a number of milliseconds since 1970/01/01
  - "isoDate" attribute: date as an international date format "YYYY-MM-DD"

An hyperlink is denoted with 2 properties:

- **href** : a URI to the resource
- **name**: a name of the hyperlink

Each property specifies value occurrences:

- **0..1**: an optional item
- **1**: a mandatory item
- **0..***: an array of items or an empty array
- **1..***: an array of items

Collections

Some URIs refer to a collection of resources.A resource can refer itself a collection of strings (see technologies)

Collections are ordered according to the items:

Collections	Order
List of items with names	Alphabetic order of names
List of snapshots items	Numerical order of snapshots, from the most recent snapshot to the less recent one
List of configuration items	Alphabetic order of keys

Configuration

REST API relies on Spring Boot configuration files and own configuration files.

Configuration files

All configuration files settings are described in documentation related to AAD which embeds REST API.

File	WAR location	ZIP location	Description
application.properties	WEB-INF/classes	zip root	Spring Boot application main configuration file Contains: • global variables • Tomcat configuration • data source configuration (each datasource is a link to a RDBMS server) • security configuration (authentication modes, configuration for LDAP, SAML, Api Key, SSL) • Jira configuration • Report Generator configuration
authorizations.xml			Authorizations applied for Measurement Service / Dashboard Service schemas.
domains.properties			REST API Domain configuration, linked to a data sources and a schema.
license.key			A license key required to access Dashboard Services schemas.
license.xml			Authorizations applied for Dashboard Services schema access in case of a restricted license.
log4j2-spring.xml			Log settings.
roles.xml			Assign a role to a user or group ("default authentication" or LDAP mode). Available roles are: • ADMIN • QUALITY_MANAGER (Action Plan - AED) • EXCLUSION_MANAGER (Violations Exclusion - AED)
user.properties			Manage "default authentication" mode users/groups.

Data Source/Domain

We can define several domains on each RDBMS Data Source.

Following names are reserved and cannot be used as a domain name

- `ping (deprecated, see user/ping)`
- `key (deprecated, see server/key)`
- `login (deprecated, see user/login)`
- `logout (deprecated, see user/logout)`
- `reload (deprecated, see server/reload)`
- `server`
- `user`

Security

For GET actions, REST API filters data according to the user authorizations (see Dashboards documentation for more details on the configuration).

- Results, list of applications, list of modules, list of technologies, list of tags are filtered
- Domain resources and System resources are not filtered, even if the user does not have authorization on any application of this resource
- Categories are not filtered (as they do not depend on applications)
- Requesting a non authorized application returns an HTTP 403 exception (Forbidden)
- Requesting a resource on a non authorized application, such as a module or snapshot returns an HTTP 403 status (Forbidden)

PUT, DELETE, UPDATE actions on tags and categories requires an **ADMIN** privilege, otherwise an HTTP 403 status is returned.

If authorizations are changed, memory cache must be reloaded to impact connected users.

Following URLs require **ADMIN** privilege, otherwise an HTTP 403 status is returned:

- `/server/reload`
- `/server/key`

License

A license key is required to get resources on a central domain.

In a case of a restriced license, license.xml file is applied for authorizations.

There are 2 exceptions. These following URLs depends on authorizations.xml settings:

- `{Domain}/applications/{ApplicationID}/snapshots/{SnapshotID}/ifpug-functions`
- `{Domain}/applications/{ApplicationID}/snapshots/{SnapshotID}/ifpug-functions-evolution`

HTTP Protocol

Authentication

Authentication is based on basic authentication on server side.

Login

Prior to any request, REST client must authenticate on behalf of the current end-user, using the "login" request. This request must contain an HTTP header containing the credentials UserName:Password encoded in base 64.

```
GET /.../rest/login HTTP/1.1
Authorization: Basic Y2FzdDpjYXN0
```

- If credentials are valid then the server replies: **HTTP/1.1 200 OK**
- If credentials are invalid then the server replies: **HTTP/1.1 401 Unauthorized**

Note: a Set-Cookie HTTP header is sent back from the server in the first server response

Logout

The following request closes the current session and replies "HTTP/1.1 401 Unauthorized"

```
GET /.../rest/logout HTTP/1.1
```

Test

To test if current client can access to the server, use the "ping" request

```
GET /.../rest/user/ping HTTP/1.1
```

Client Cache Management

In order to take benefit of navigator response cache on client side, each response is replied with a "ETag" directive header, using the cache loading date:

Example

```
ETag: "application/json, text/javascript, */*; q=0.01:Mon Aug 19 17:04:16 CEST 2013"
```

Note: No cache control directive is sent to the client.

Upon request with directive "If-None-Match", the server compares the provided date with the cache loading date.

If they are equal, then the server replies with HTTP status 304:

Example

```
HTTP/1.1 304 Not Modified
```

API key

The API key is an alternative to the Basic Authentication mechanism.

A key is a secret text (it can be unique random text) stored in the application.properties file:

Example

```
security.apikey=22b56696-7d59-11e9-8f9e-2a86e4085a59
```

Any client can use this key to authenticate any user, with a request containing two HTTP headers:

- X-API-KEY: a key matching the one in the application.properties file
- X-API-USER: a user name to get a role and authorizations

For example we can call the "server/reload" web service to refresh the REST API memory cache as follow:

Example

```
C:>curl -k https://localhost:8080/Dashboard-WebService/rest/server/reload -H "X-API-KEY: 22b56696-7d59-11e9-8f9e-2a86e4085a59" -H "X-API-USER: admin" -v
```

The API KEY is required to bypass the SAML Single-Sign On protocol, for non-browser clients, and the Report Generator executed in a standalone mode or integrated mode.

Note that the user is granted with "ROLE_ADMIN" role when the API KEY is used.

X-Client HTTP header

This header can be set by any REST API client in order to identify the effective client in the audit trail, for example:

Example

```
C:>curl -k https://localhost:8080/Dashboard-WebService/rest/server -H "X-Client: CURL" -H "X-API-KEY: 22b56696-7d59-11e9-8f9e-2a86e4085a59" -H "X-API-USER: admin" -v
```

Query Parameters Encoding

Query parameters must be encoded, especially if they contain some special characters such like : space, #, +, etc.

For example:

The following URL

http://localhost:8080/Dashboard-WebService/rest/AAD/results?technologies=(C++)

must be encoded as follow:

http://localhost:8080/Dashboard-WebService/rest/AAD/results?technologies=(C%2B%2B)

Note that the Simple REST Client (default.html) encodes parameters.

Errors

HTTP Status	Code	Example of messages	Circumstances
400 Bad Request	1	Cannot process this request	1. This URI is not recognized, for example: "*GET D/applications/A/snapshots/S/action-plan/triggers*" is requested with S not being a snapshot of application A 2. In case of a GET operation, there is an inappropriate/unsafe value for a query parameter, or URI fragment. For example: a. "*GET AAD/applications/A/snapshots/S/action-plan/triggers*" - is requested with S not being a last snapshot b. "*GET AAD/applications/A/results?snapshots=(-1)&snapshot-ids=(4,5)*" - snapshots and snapshot-ids parameters are exclusive
	2	Incorrect parameter This request may return too many items. Query parameter 'rule-pattern=$any' is not allowed for CSV	This URI contains a query-string with an unexpected parameter value, for example: "*GET AAD/applications/A/snapshots/S/action-plan/issues?startRow=1&nbRows=-1*"- nbRows has an incorrect value.
	3	Resource not implemented for this domain	DBMS schema of this domain is too old, an upgrade is required to use this feature
	4	Invalid content	In case of a PUT, POST, DELETE operation 1. there is a JSON syntax error 2. there is an incorrect property name, property type, property value 3. there is an inappropriate/unsafe property value 4. a property is missing
403 Forbidden	1	You are not allowed to access this data.	A user has no access to these data according to the authorizations.xml file or license.xml file
404 Not Found	1	Resource not found	This URI contains IDs matching no resource.
406 Not Acceptable	1	The requested media type is not appropriate for this resource	Representation Negociation failed. In other words, 'Accept' HTTP Header does not match any expected content type: 'application/json', 'text/csv', ... Most of resources support a single representation (i.e. a single media type)
500 Internal Server Error	1	Run-time exception cause and call stack	Unexpected exception
	20	`{ "cause": "...", ...`	Report Generator generation has failed.
503 Service Unavailable	1	Server status: LOADING	Domains are being loaded into memory cache
	2	DBMS connections has failed or all domains loading have been aborted	DBMS is not started, DBMS resources are not available, or datasource configuration in application.properties is not correct
	20	`{ "cause": "...", ...`	Report Generator configuration settings is not correct. See Report Service.

JSON response example:

Example

```
{"code": 1, "message": "Cannot process this request"}
```

Note that unrecognized parameters in a URL (for example "application" spelt incorrectly in the URL "*AAD/results?quality-indicators=(quality-rules)&select=(evolutionSummary,violationRatio)&snapshots=2017-12-19&aplications=TEST_APPLICATION*") will cause the unrecognized parameter to be ignored. The remainder of the URL (assuming that it is correct) will return results. In the example given, the REST API would return results for all applications. In other words, no error code will be returned.