BLACKDUCK | Hub

Getting Started with the Hub SDK

Version 4.4.0

This edition of the *Getting Started with the Hub SDK* refers to version 4.4.0 of the Black Duck Hub.

This document created or updated on Thursday, December 14, 2017.

Please send your comments and suggestions to:

Black Duck Software, Incorporated
800 District Avenue, Suite 201
Burlington, MA 01803-5061 USA

Contents

The Hub documentation

The documentation for the Hub consists of online help and these documents:

Title	File	Description
Release Notes	release_notes_bd_hub.pdf	Contains information about the new and improved features, resolved issues, and known issues in the current and previous releases.
Installing Hub using Docker Compose	hub_install_compose.pdf	Contains information about installing and upgrading the Hub using Docker Compose.
Installing Hub using Docker Swarm	hub_install_swarm.pdf	Contains information about installing and upgrading the Hub using Docker Swarm.
Installing Hub using Kubernetes	hub_install_kubernetes.pdf	Contains information about installing and upgrading the Hub using Kubernetes.
Installing Hub using OpenShift	hub_install_openshift.pdf	Contains information about installing and upgrading the Hub using OpenShift.
Getting Started	hub_getting_started.pdf	Provides first-time users with information on using the Hub.
Scanning Best Practices	hub_scanning_best_practices.pdf	Provides best practices for scanning.
Getting Started with the Hub SDK	getting_started_hub_sdk.pdf	Contains overview information and a sample use case.
Report Database	report_db_bd_hub.pdf	Contains information on using the report database.

Hub integration documentation can be found on Confluence.

Training

Black Duck Academy is a one-stop resource for all your Black Duck education needs. It provides you with 24x7 access to online training courses and how-to videos.

New videos and courses are added monthly.

At Black Duck Academy, you can:

- Learn at your own pace.
- Review courses as often as you wish.
- Take assessments to test your skills.
- Print certificates of completion to showcase your accomplishments.

Learn more at https://www.blackducksoftware.com/services/training

View the full catalog of courses and try some free courses at https://academy.blackducksoftware.com

When you are ready to learn, log in or sign up for an account: https://academy.blackducksoftware.com

Customer Success Community

The Black Duck Customer Success Community is our primary online resource for customer support, solutions and information. The Customer Success Community allows users to quickly and easily open support cases and monitor progress, learn important product information, search a knowledgebase, and gain insights from other Black Duck customers. The many features included in the Customer Success Community center around the following collaborative actions:

- Connect – Open support cases and monitor their progress, as well as, monitor issues that require Engineering or Product Management assistance
- Learn – Insights and best practices from other Black Duck product users to allow you to learn valuable lessons from a diverse group of industry leading companies. In addition, the Customer Hub puts all the latest product news and updates from Black Duck at your fingertips, helping you to better utilize our products and services to maximize the value of open source within your organization.
- Solve – Quickly and easily get the answers you're seeking with the access to rich content and product knowledge from Black Duck experts and our Knowledgebase.
- Share – Collaborate and connect with Black Duck staff and other customers to crowdsource solutions and share your thoughts on product direction.

Access the Customer Success Community. If you do not have an account or have trouble accessing the system, please send an email to communityfeedback@blackducksoftware.com or call us at +1 781.891.5100 ext. 5.

To see all the ways you can interact with Black Duck Support, visit: https://www.blackducksoftware.com/support/contact-support.

Chapter 1: About the Hub SDK

The Black Duck Hub REST APIs provide a standard interface for interacting with the Hub.

REST APIs provide access to resources (data entities) via URI paths. To use a REST API, your application will make a HTTP request and parse the response. The methods used by your application will be the standard HTTP methods of GET, POST, PUT and DELETE. REST APIs operate over HTTP(S) making it easy to use with any programming language or framework. The input and output format for the Hub REST APIs is JSON.

Important: Black Duck does not support REST APIs that begin with "v1" or "internal".

Viewing and Testing the REST APIs

The Black Duck Hub provides online access and documentation to the REST servers and individual REST APIs. After logging in to the Hub application, you can enter the following url to access this information:

http://<your Hub server url>/api.html

Each supported REST API is documented with the required input parameters and expected response. A "Try it out" button allows you to do interactive testing of the API.

Note: The API page uses Swagger to document the Hub APIs. Swagger documentation should not be considered part of the Hub REST APIs and can change without warning or backward compatibility.

The following example shows the component-version REST API call, which returns component information:

```
GET /api/components/{componentId}/versions/{versionId}
```

where `{componentID`} and `{versionID}` are the internal Hub IDs for the OSS component in the Black Duck KB.

Curl

```
curl -X GET --header "Accept: application/json" "https://<HUB SERVER>/api/components/b2d3d36c-15d3-4e1a-bef5-
8e907f107048/versions/35f6d76e-e4dd-4b87-bae5-133da093dcd2"
```

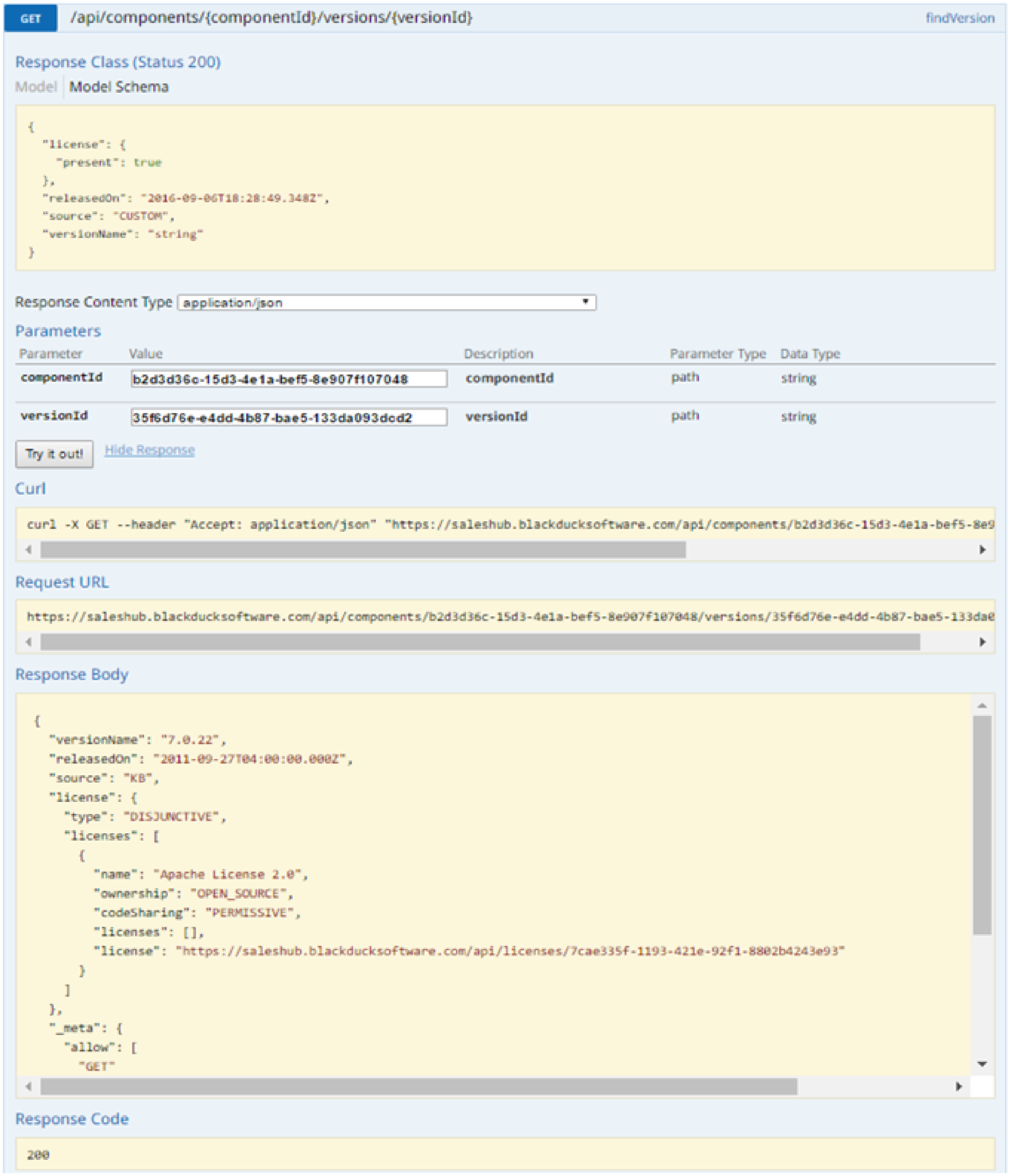
Request URL

```
https://<HUB_SERVER>/api/components/b2d3d36c-15d3-4e1a-bef5-8e907f107048/versions/35f6d76e-e4dd-4b87-bae5-133da093dcd2
```

Response Body

```
{
 "versionName": "7.0.22",
 "releasedOn": "2011-09-27T04:00:00.000Z",
 "source": "KB",
 "license": {
  "type": "DISJUNCTIVE",
  "licenses": [
   {
    "name": "Apache License 2.0",
    "ownership": "OPEN_SOURCE",
    "codeSharing": "PERMISSIVE",
    "licenses": [],
    "license": "https://saleshub.blackducksoftware.com/api/licenses/7cae335f-1193-421e-92f1-8802b4243e93"
   }

  ]
 },
```

The equivalent API call can also be made using the API GUI, for example:



GET /api/components/{componentId}/versions/{versionId} findVersion

Response Class (Status 200)

Model | Model Schema

```
{
  "license": {
    "present": true
  },
  "releasedOn": "2016-09-06T18:28:49.348Z",
  "source": "CUSTOM",
  "versionName": "string"
}
```

Response Content Type: application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
componentId	b2d3d36c-15d3-4e1a-bef5-8e907f107048	componentId	path	string
versionId	35f6d76e-e4dd-4b87-bae5-133da093dcd2	versionId	path	string

Try it out! Hide Response

Curl

```
curl -X GET --header "Accept: application/json" "https://saleshub.blackducksoftware.com/api/components/b2d3d36c-15d3-4e1a-bef5-8e9
```

Request URL

```
https://saleshub.blackducksoftware.com/api/components/b2d3d36c-15d3-4e1a-bef5-8e907f107048/versions/35f6d76e-e4dd-4b87-bae5-133da0
```

Response Body

```
{
  "versionName": "7.0.22",
  "releasedOn": "2011-09-27T04:00:00.000Z",
  "source": "KB",
  "license": {
    "type": "DISJUNCTIVE",
    "licenses": [
      {
        "name": "Apache License 2.0",
        "ownership": "OPEN_SOURCE",
        "codeSharing": "PERMISSIVE",
        "licenses": [],
        "license": "https://saleshub.blackducksoftware.com/api/licenses/7cae335f-1193-421e-92f1-8802b4243e93"
      }
    ]
  },
  "_meta": {
    "allow": [
      "GET"
```

Response Code

```
200
```

Authentication

The Black Duck Hub REST APIs currently support session authentication. In order to access the REST APIs, you must first authenticate with the Hub application. For example, the following curl command shows how to authenticate with the Hub application prior to making the REST API calls:

```
curl -X POST --data 'j_username=username&j_password=password' -i http://<your
Hub server url>/j_spring_security_check
```

where the username and password in bold are the credentials for logging in to your Hub instance.

Chapter 2: Example of using the Hub SDK

Use Case: I'm interested in using the REST APIs to find risk counts for my project "Application1" version "1.0".

Step 1: Authentication

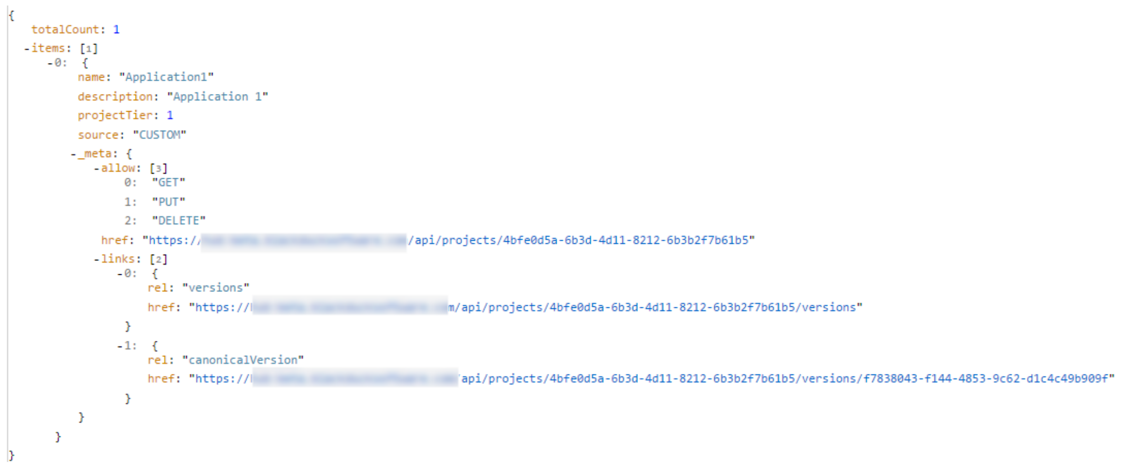
You must first authenticate with the Hub application. For example, the following curl command shows how to authenticate with the Hub application prior to making the REST API calls:

```
curl -X POST --data 'j_username=username&j_password=password' -i http://<your Hub server url>/j_spring_security_check
```

Note: the username and password in bold are the credentials for logging in to your Hub instance.

Step 2: Get the Application1 project

https://<Hub server url>:443/api/projects?q=name%3AApplication1



```
{
   totalCount: 1
  -items: [1]
     -0:  {
           name: "Application1"
           description: "Application 1"
           projectTier: 1
           source: "CUSTOM"
          -_meta: {
             -allow: [3]
                 0:  "GET"
                 1:  "PUT"
                 2:  "DELETE"
              href: "https://[redacted]/api/projects/4bfe0d5a-6b3d-4d11-8212-6b3b2f7b61b5"
             -links: [2]
                -0:  {
                      rel: "versions"
                      href: "https://[redacted]m/api/projects/4bfe0d5a-6b3d-4d11-8212-6b3b2f7b61b5/versions"
                  }
                -1:  {
                      rel: "canonicalVersion"
                      href: "https://[redacted]api/projects/4bfe0d5a-6b3d-4d11-8212-6b3b2f7b61b5/versions/f7838043-f144-4853-9c62-d1c4c49b909f"
                  }
             }
       }
}
```

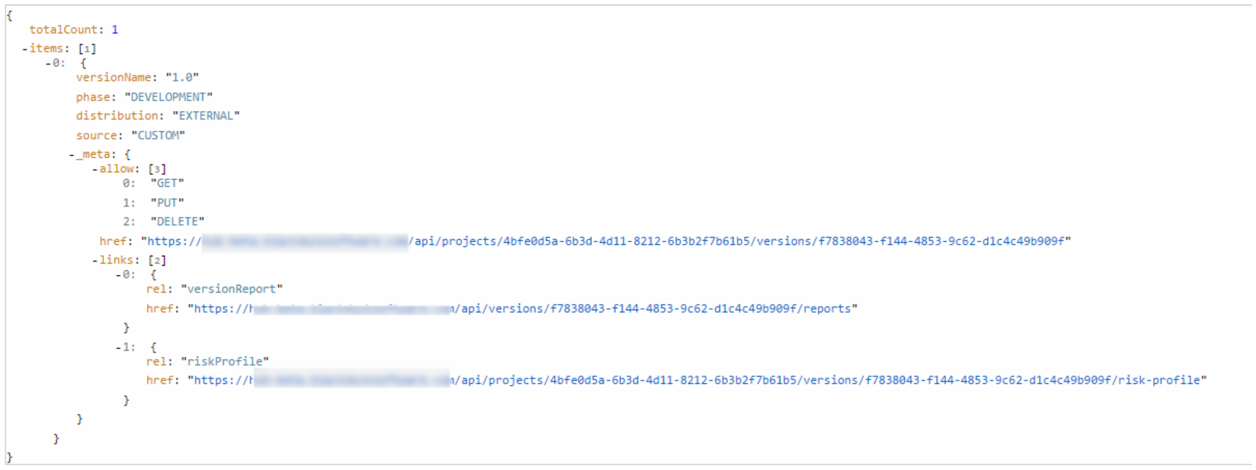
Included in the JSON response are all your available http calls. Notice you can call "versions" or "canonicalVersion".

Step 3: Get the Application1 project versions

Get the versions of Application1 project.

Using the links provided in the JSON response in step 1 I can use the following to get my project versions:

https://<Hub server url>/api/projects/4bfe0d5a-6b3d-4d11-8212-6b3b2f7b61b5/versions



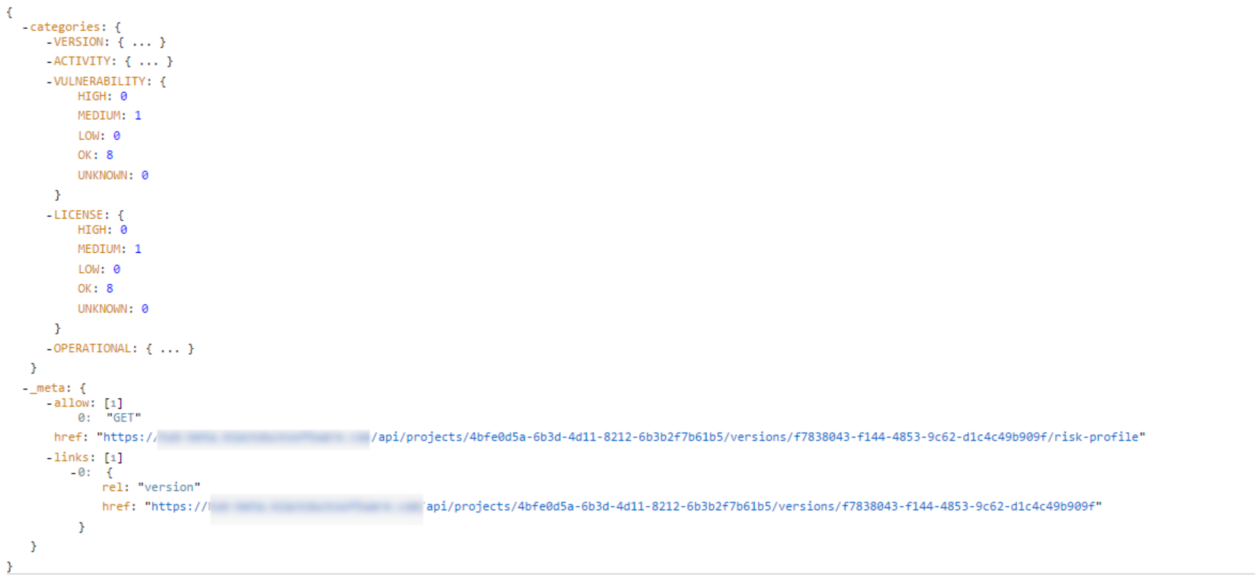
```
{
   totalCount: 1
  -items: [1]
     -0:  {
           versionName: "1.0"
           phase: "DEVELOPMENT"
           distribution: "EXTERNAL"
           source: "CUSTOM"
          -_meta: {
             -allow: [3]
                 0:  "GET"
                 1:  "PUT"
                 2:  "DELETE"
              href: "https://[redacted]/api/projects/4bfe0d5a-6b3d-4d11-8212-6b3b2f7b61b5/versions/f7838043-f144-4853-9c62-d1c4c49b909f"
             -links: [2]
                -0:  {
                       rel: "versionReport"
                       href: "https://[redacted]/api/versions/f7838043-f144-4853-9c62-d1c4c49b909f/reports"
                    }
                -1:  {
                       rel: "riskProfile"
                       href: "https://[redacted]/api/projects/4bfe0d5a-6b3d-4d11-8212-6b3b2f7b61b5/versions/f7838043-f144-4853-9c62-d1c4c49b909f/risk-profile"
                    }
           }
       }
}
```

Included in the JSON response are all your available http calls. Notice you can call “riskProfile”.

Step 4: Get the Risk Profile for “1.0” version

Using the links in the JSON response from above I can call the following to get the risk profile:

https://<Hub server url>/api/projects/4bfe0d5a-6b3d-4d11-8212-6b3b2f7b61b5/versions/f7838043-f144-4853-9c62-d1c4c49b909f/risk-profile



```
{
  -categories: {
     -VERSION: { ... }
     -ACTIVITY: { ... }
     -VULNERABILITY: {
          HIGH: 0
          MEDIUM: 1
          LOW: 0
          OK: 8
          UNKNOWN: 0
      }
     -LICENSE: {
          HIGH: 0
          MEDIUM: 1
          LOW: 0
          OK: 8
          UNKNOWN: 0
      }
     -OPERATIONAL: { ... }
   }
  -_meta: {
     -allow: [1]
          0:  "GET"
      href: "https://[redacted]/api/projects/4bfe0d5a-6b3d-4d11-8212-6b3b2f7b61b5/versions/f7838043-f144-4853-9c62-d1c4c49b909f/risk-profile"
     -links: [1]
        -0:  {
              rel: "version"
              href: "https://[redacted]api/projects/4bfe0d5a-6b3d-4d11-8212-6b3b2f7b61b5/versions/f7838043-f144-4853-9c62-d1c4c49b909f"
          }
   }
}
```

The JSON response includes all risk counts for Version, Activity, Vulnerability, License, and Operational risk. The user can now use those values for whatever external activity they’d like to do (fail a build, create bug tracking tickets, and so on)

Chapter 3: CSRF security

Cross-site request forgeries (CSRF) are types of attacks which perform unwanted actions on a web application without the user's intent. This type of vulnerability happens on web applications which use cookies to identify the user. For example, the attacker uses the same cookies to make a fraudulent request, and attempts to lure the user to click a fraudulent link. If the user clicks the link after being authenticated, the fraudulent request performs any actions available to the logged-in user, but without the knowledge of the user.

The forged request works because browser requests automatically include all credentials associated with the web application, such as the logged-in user's session information in cookies. The server cannot distinguish between the forged request from the user and the legitimate request form the user.

Other names

CSRF attacks are also known by other names; the main difference is the technique used to perform the attack. The names include XSRF, Sea Surf, Session Riding, and Hostile Liking.

CSRF security in The Hub

Black Duck Hub version 4.0.0 now features improved security for attempted cross-site request forgeries (CSRF). The Hub uses a Synchronized Token Pattern (STP) to address CSRF. This is a technique where a secret and unique token value is passed along with every request through form data or the header. This token is then verified on the server side. If the tokens do not match, then the request is ignored, and a *403 forbidden* error displays. The unique token is generated for each session, not for each request. The same unique token is used for the duration of the session, and a new token is generated for each new session. The token is destroyed when its session is destroyed. The following is an example of the CSRF format:

```
headerName = X-CSRF-TOKEN

parameterName = _csrf

token = 484dcea0-82a7-4f3e-9158-f80513ed86f9
```

When making a REST call to authenticate the user, a POST request (login request) is made to the URL `/j_spring_security_check`. The CSRF information is returned in the response header. The following is an example of the *login request header*:

```
POST /j_spring_security_check HTTP/1.1

Host: qa-hub21.dc1.lan

Connection: keep-alive

Content-Length: 40
```

```
Origin: https:

//qa-hub21.dc1.lan

User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_5)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110
Safari/537.36

Content-Type: application/x-www-form-urlencoded; charset=UTF-8

Accept: */*

X-Requested-With: XMLHttpRequest

X-FirePHP-Version: 0.0.6

Referer: https://qa-hub21.dc1.lan/

Accept-Encoding: gzip, deflate, br

Accept-Language: en-US,en;q=0.8,ja;q=0.6,cs;q=0.4

Cookie: JSESSIONID=5AB917FDED71241D77EB2EAC4C364DA0
```

The following is an example of the returned *login response header*. The CSRF token is highlighted.

```
HTTP/1.1 204

Server: nginx/1.10.3

Date: Wed, 14 Jun 2017 16:27:50 GMT

Connection: keep-alive

X-CSRF-HEADER: X-CSRF-TOKEN

X-CSRF-PARAM: _csrf

X-CSRF-TOKEN: 484dcea0-82a7-4f3e-9158-f80513ed86f9

Set-Cookie:
JSESSIONID=EC4F20CE9ADAACD02421803421B75D9B;path=/;HttpOnly

X-Content-Type-Options: nosniff

X-XSS-Protection: 1; mode=block

Cache-Control: no-cache, no-store, max-age=0, must-revalidate

Pragma: no-cache

Expires: 0

X-Frame-Options: SAMEORIGIN
```

To add the CSRF security header to the request:

1. Copy the CSRF token from the login response header, and include the CSRF token in each subsequent `POST`, `PUT`, `PATCH` and `DELETE` request headers.

The following is a sample request with the CSRF token passed in using the request header:

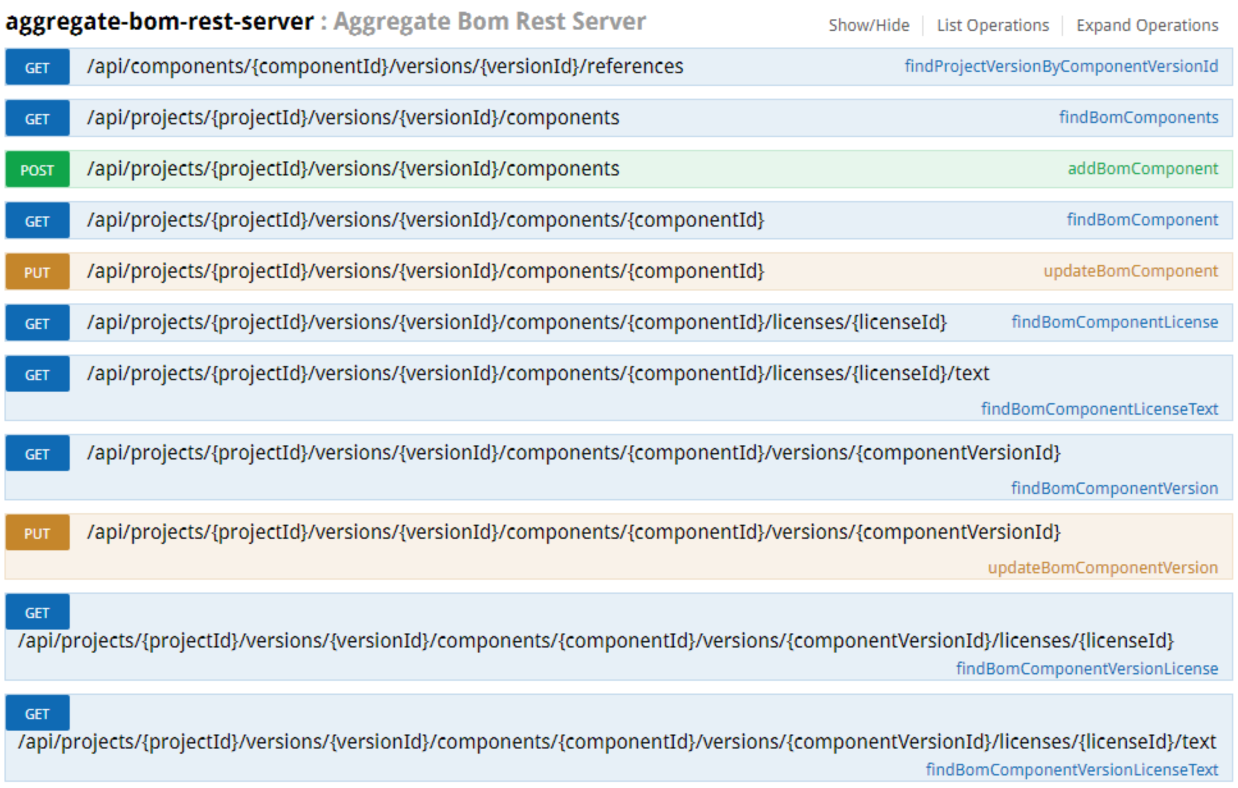
```
POST /api/v1/projects HTTP/1.1
```

```
Host: qa-hub21

Connection: keep-alive

Content-Length: 180

Origin: https://qa-hub21

X-CSRF-TOKEN: 484dcea0-82a7-4f3e-9158-f80513ed86f9

User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_5)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110
Safari/537.36

Content-Type: application/json

Accept: application/json, text/javascript, */*; q=0.01

X-Requested-With: XMLHttpRequest

X-FirePHP-Version: 0.0.6

Referer: https://qa-hub21/ui/projects/view:create/action:modal

Accept-Encoding: gzip, deflate, br

Accept-Language: en-US,en;q=0.8,ja;q=0.6,cs;q=0.4

Cookie: JSESSIONID=3515A2241A424B3523436B354823F86D
```

Note: You must use the CSRF token from your specific HTTP session in your request. Any integration with a Hub 4.0.0 (or higher) server must include the token.

Chapter 4: API list

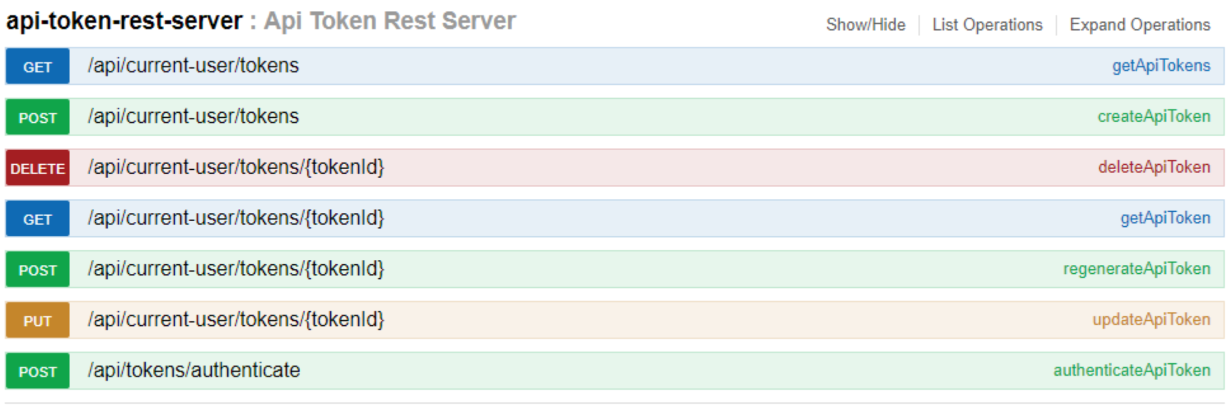
Each supported REST API is documented with the required input parameters. Below is the API list for the Hub 4.4.0:



aggregate-bom-rest-server : Aggregate Bom Rest Server

Show/Hide | List Operations | Expand Operations

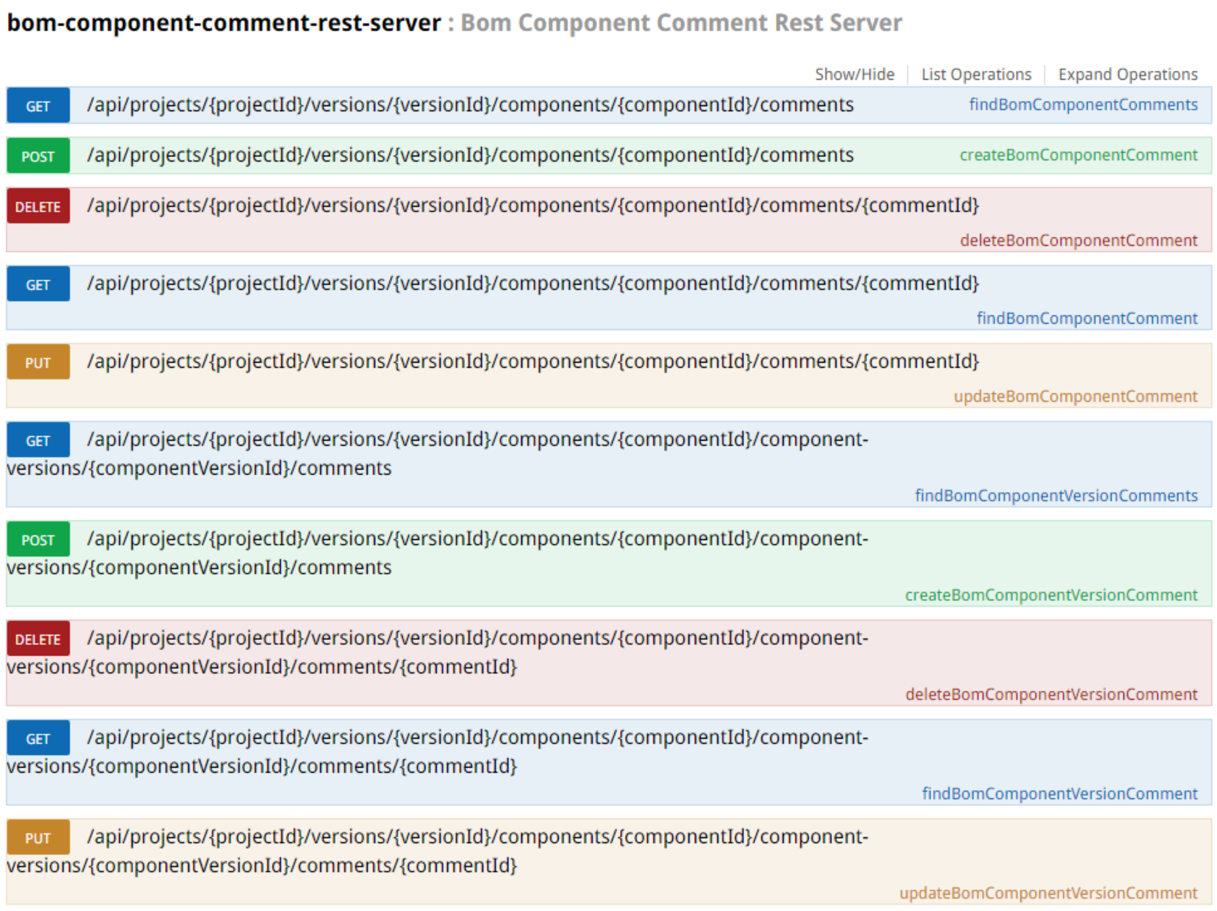
GET	/api/components/{componentId}/versions/{versionId}/references	findProjectVersionByComponentVersionId
GET	/api/projects/{projectId}/versions/{versionId}/components	findBomComponents
POST	/api/projects/{projectId}/versions/{versionId}/components	addBomComponent
GET	/api/projects/{projectId}/versions/{versionId}/components/{componentId}	findBomComponent
PUT	/api/projects/{projectId}/versions/{versionId}/components/{componentId}	updateBomComponent
GET	/api/projects/{projectId}/versions/{versionId}/components/{componentId}/licenses/{licenseId}	findBomComponentLicense
GET	/api/projects/{projectId}/versions/{versionId}/components/{componentId}/licenses/{licenseId}/text	findBomComponentLicenseText
GET	/api/projects/{projectId}/versions/{versionId}/components/{componentId}/versions/{componentVersionId}	findBomComponentVersion
PUT	/api/projects/{projectId}/versions/{versionId}/components/{componentId}/versions/{componentVersionId}	updateBomComponentVersion
GET	/api/projects/{projectId}/versions/{versionId}/components/{componentId}/versions/{componentVersionId}/licenses/{licenseId}	findBomComponentVersionLicense
GET	/api/projects/{projectId}/versions/{versionId}/components/{componentId}/versions/{componentVersionId}/licenses/{licenseId}/text	findBomComponentVersionLicenseText



api-token-rest-server : Api Token Rest Server

Show/Hide | List Operations | Expand Operations

GET	/api/current-user/tokens	getApiTokens
POST	/api/current-user/tokens	createApiToken
DELETE	/api/current-user/tokens/{tokenId}	deleteApiToken
GET	/api/current-user/tokens/{tokenId}	getApiToken
POST	/api/current-user/tokens/{tokenId}	regenerateApiToken
PUT	/api/current-user/tokens/{tokenId}	updateApiToken
POST	/api/tokens/authenticate	authenticateApiToken



bom-component-comment-rest-server : Bom Component Comment Rest Server

Show/Hide | List Operations | Expand Operations

GET /api/projects/{projectId}/versions/{versionId}/components/{componentId}/comments findBomComponentComments

POST /api/projects/{projectId}/versions/{versionId}/components/{componentId}/comments createBomComponentComment

DELETE /api/projects/{projectId}/versions/{versionId}/components/{componentId}/comments/{commentId} deleteBomComponentComment

GET /api/projects/{projectId}/versions/{versionId}/components/{componentId}/comments/{commentId} findBomComponentComment

PUT /api/projects/{projectId}/versions/{versionId}/components/{componentId}/comments/{commentId} updateBomComponentComment

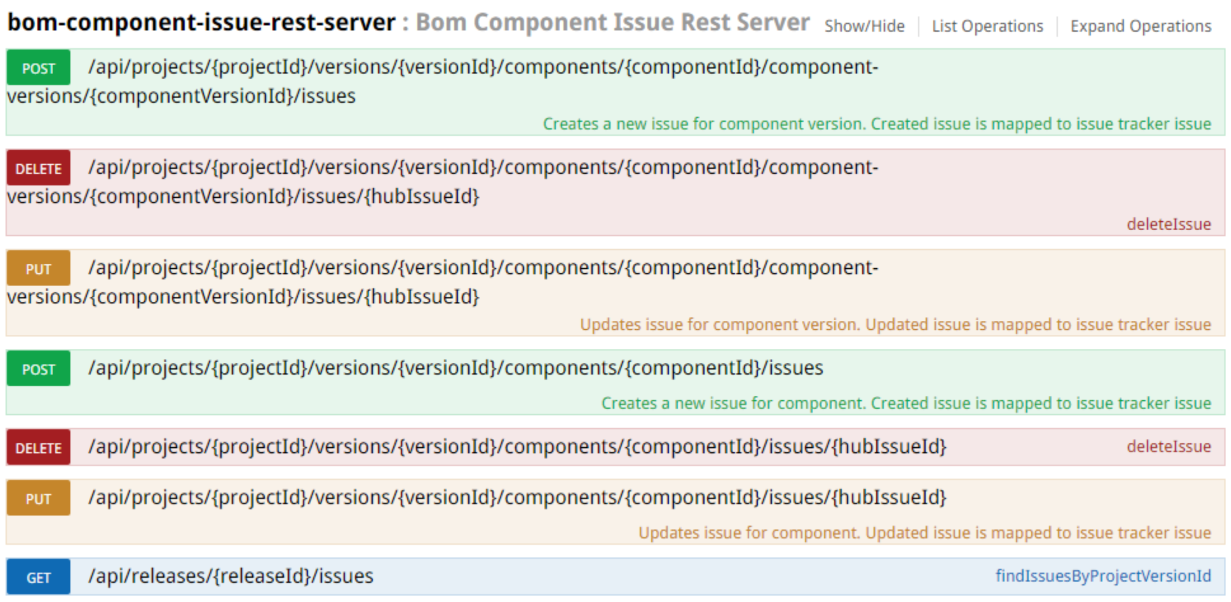
GET /api/projects/{projectId}/versions/{versionId}/components/{componentId}/component-versions/{componentVersionId}/comments findBomComponentVersionComments

POST /api/projects/{projectId}/versions/{versionId}/components/{componentId}/component-versions/{componentVersionId}/comments createBomComponentVersionComment

DELETE /api/projects/{projectId}/versions/{versionId}/components/{componentId}/component-versions/{componentVersionId}/comments/{commentId} deleteBomComponentVersionComment

GET /api/projects/{projectId}/versions/{versionId}/components/{componentId}/component-versions/{componentVersionId}/comments/{commentId} findBomComponentVersionComment

PUT /api/projects/{projectId}/versions/{versionId}/components/{componentId}/component-versions/{componentVersionId}/comments/{commentId} updateBomComponentVersionComment



bom-component-issue-rest-server : Bom Component Issue Rest Server Show/Hide | List Operations | Expand Operations

POST /api/projects/{projectId}/versions/{versionId}/components/{componentId}/component-versions/{componentVersionId}/issues Creates a new issue for component version. Created issue is mapped to issue tracker issue

DELETE /api/projects/{projectId}/versions/{versionId}/components/{componentId}/component-versions/{componentVersionId}/issues/{hubIssueId} deleteIssue

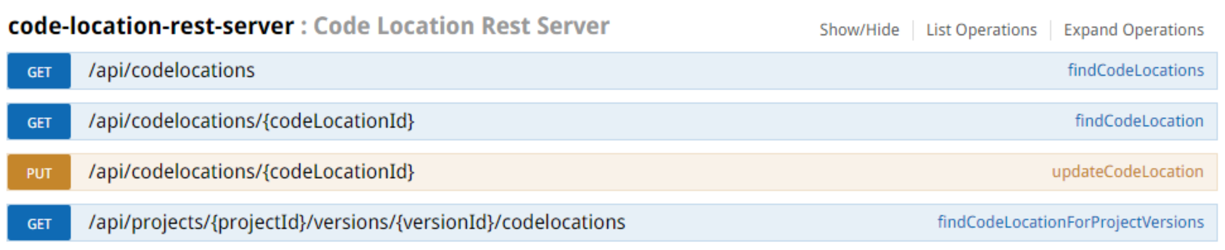
PUT /api/projects/{projectId}/versions/{versionId}/components/{componentId}/component-versions/{componentVersionId}/issues/{hubIssueId} Updates issue for component version. Updated issue is mapped to issue tracker issue

POST /api/projects/{projectId}/versions/{versionId}/components/{componentId}/issues Creates a new issue for component. Created issue is mapped to issue tracker issue

DELETE /api/projects/{projectId}/versions/{versionId}/components/{componentId}/issues/{hubIssueId} deleteIssue

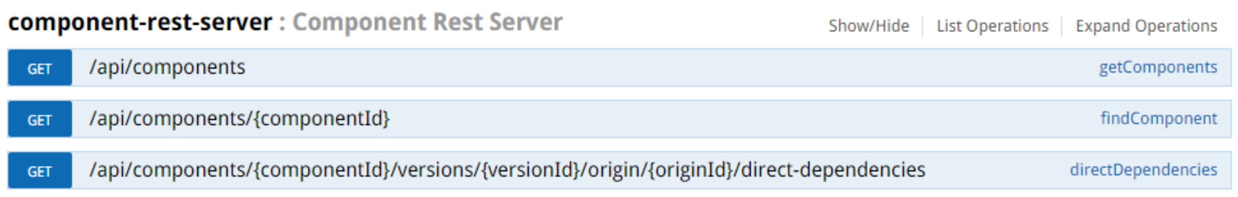
PUT /api/projects/{projectId}/versions/{versionId}/components/{componentId}/issues/{hubIssueId} Updates issue for component. Updated issue is mapped to issue tracker issue

GET /api/releases/{releaseId}/issues findIssuesByProjectVersionId



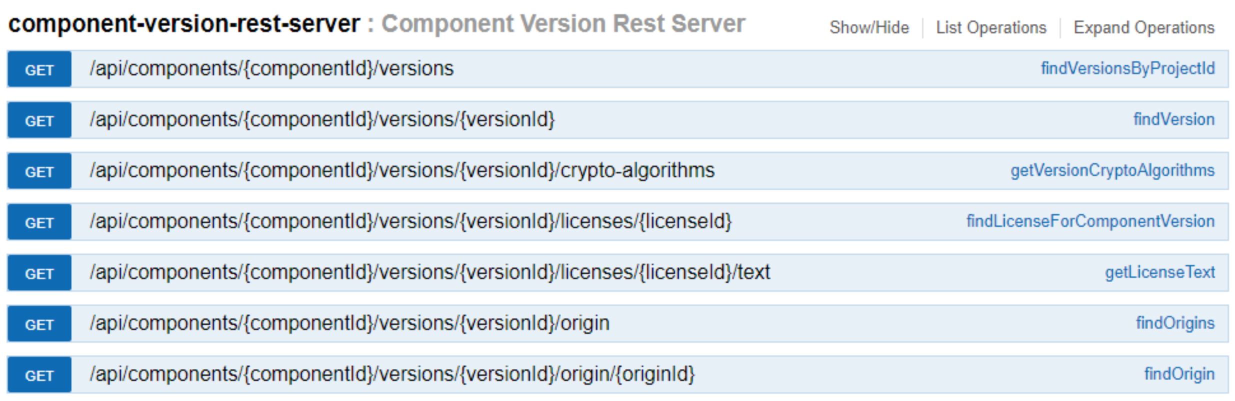
code-location-rest-server : Code Location Rest Server — Show/Hide | List Operations | Expand Operations

GET	/api/codelocations	findCodeLocations
GET	/api/codelocations/{codeLocationId}	findCodeLocation
PUT	/api/codelocations/{codeLocationId}	updateCodeLocation
GET	/api/projects/{projectId}/versions/{versionId}/codelocations	findCodeLocationForProjectVersions



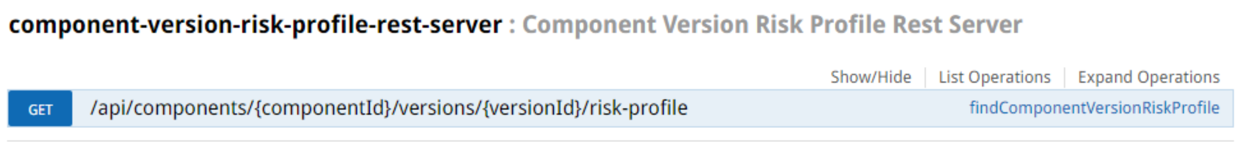
component-rest-server : Component Rest Server — Show/Hide | List Operations | Expand Operations

GET	/api/components	getComponents
GET	/api/components/{componentId}	findComponent
GET	/api/components/{componentId}/versions/{versionId}/origin/{originId}/direct-dependencies	directDependencies



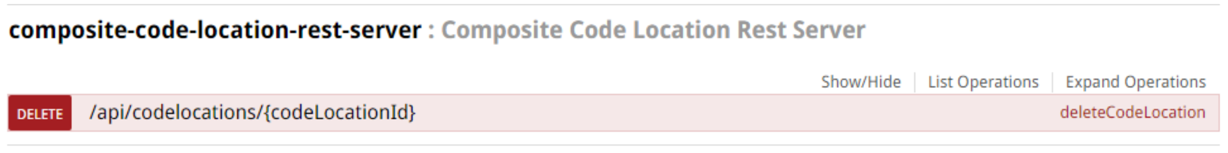
component-version-rest-server : Component Version Rest Server — Show/Hide | List Operations | Expand Operations

GET	/api/components/{componentId}/versions	findVersionsByProjectId
GET	/api/components/{componentId}/versions/{versionId}	findVersion
GET	/api/components/{componentId}/versions/{versionId}/crypto-algorithms	getVersionCryptoAlgorithms
GET	/api/components/{componentId}/versions/{versionId}/licenses/{licenseId}	findLicenseForComponentVersion
GET	/api/components/{componentId}/versions/{versionId}/licenses/{licenseId}/text	getLicenseText
GET	/api/components/{componentId}/versions/{versionId}/origin	findOrigins
GET	/api/components/{componentId}/versions/{versionId}/origin/{originId}	findOrigin



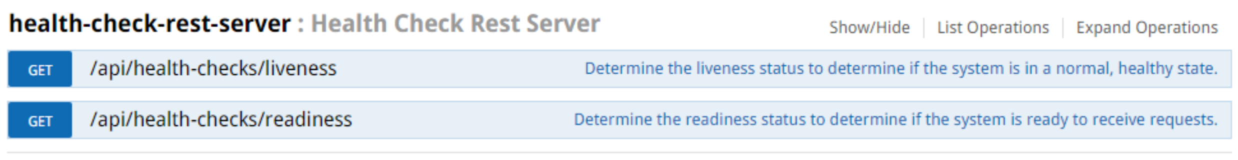
component-version-risk-profile-rest-server : Component Version Risk Profile Rest Server — Show/Hide | List Operations | Expand Operations

GET	/api/components/{componentId}/versions/{versionId}/risk-profile	findComponentVersionRiskProfile



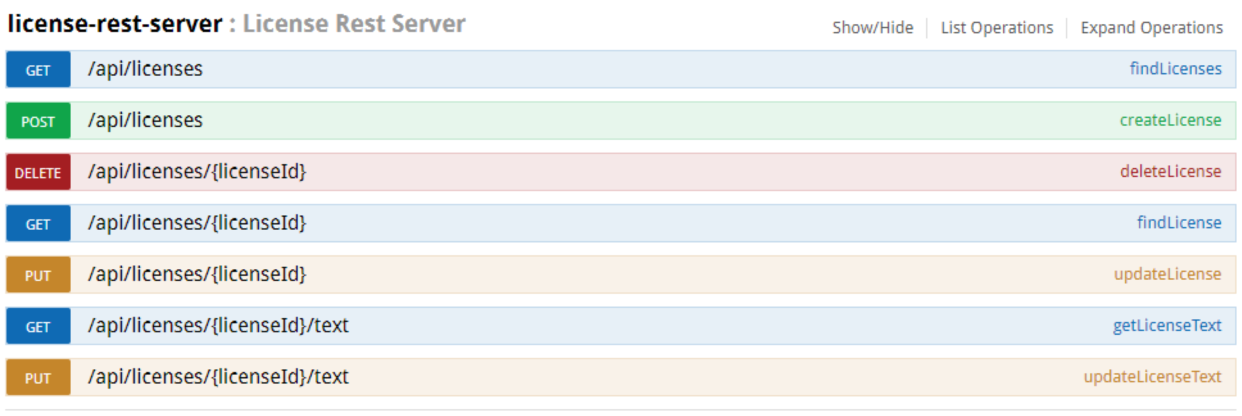
composite-code-location-rest-server : Composite Code Location Rest Server — Show/Hide | List Operations | Expand Operations

DELETE	/api/codelocations/{codeLocationId}	deleteCodeLocation



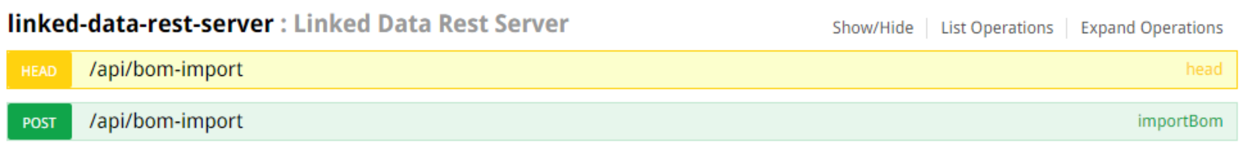
health-check-rest-server : Health Check Rest Server — Show/Hide | List Operations | Expand Operations

GET	/api/health-checks/liveness	Determine the liveness status to determine if the system is in a normal, healthy state.
GET	/api/health-checks/readiness	Determine the readiness status to determine if the system is ready to receive requests.


license-rest-server : License Rest Server

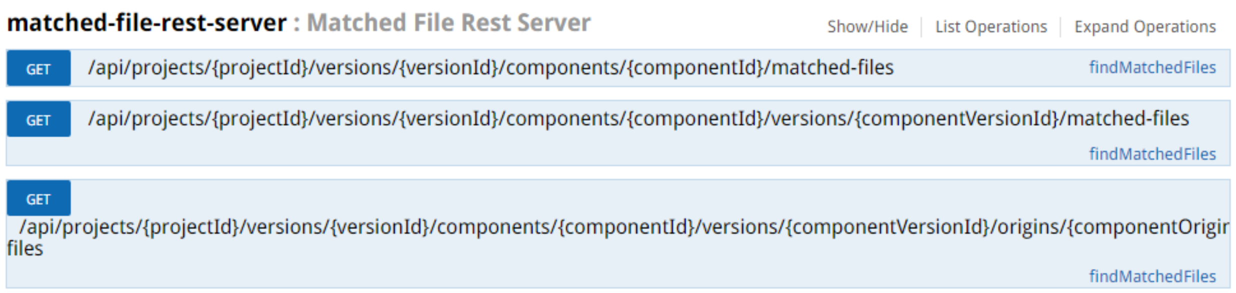
Show/Hide | List Operations | Expand Operations

GET	/api/licenses	findLicenses
POST	/api/licenses	createLicense
DELETE	/api/licenses/{licenseId}	deleteLicense
GET	/api/licenses/{licenseId}	findLicense
PUT	/api/licenses/{licenseId}	updateLicense
GET	/api/licenses/{licenseId}/text	getLicenseText
PUT	/api/licenses/{licenseId}/text	updateLicenseText


linked-data-rest-server : Linked Data Rest Server

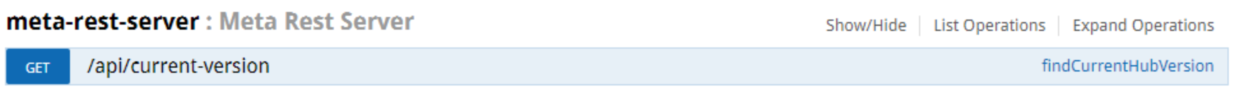
Show/Hide | List Operations | Expand Operations

HEAD	/api/bom-import	head
POST	/api/bom-import	importBom


matched-file-rest-server : Matched File Rest Server

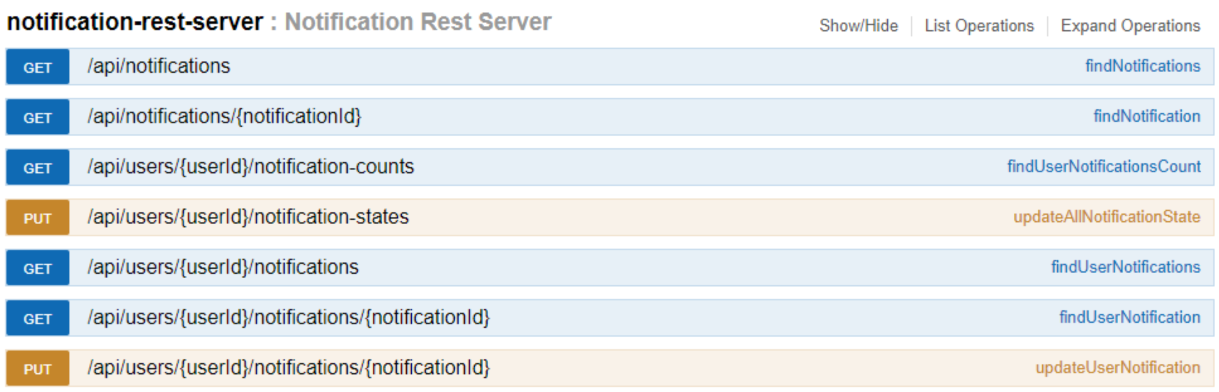
Show/Hide | List Operations | Expand Operations

GET	/api/projects/{projectId}/versions/{versionId}/components/{componentId}/matched-files	findMatchedFiles
GET	/api/projects/{projectId}/versions/{versionId}/components/{componentId}/versions/{componentVersionId}/matched-files	findMatchedFiles
GET	/api/projects/{projectId}/versions/{versionId}/components/{componentId}/versions/{componentVersionId}/origins/{componentOrigir files	findMatchedFiles


meta-rest-server : Meta Rest Server

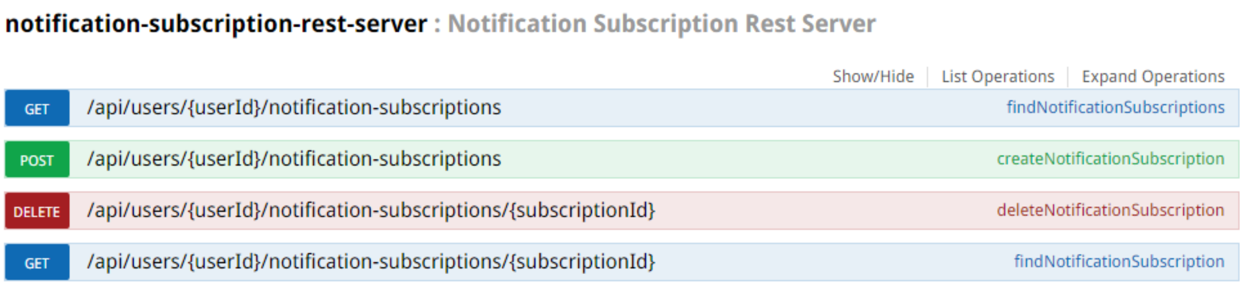
Show/Hide | List Operations | Expand Operations

GET	/api/current-version	findCurrentHubVersion


notification-rest-server : Notification Rest Server

Show/Hide | List Operations | Expand Operations

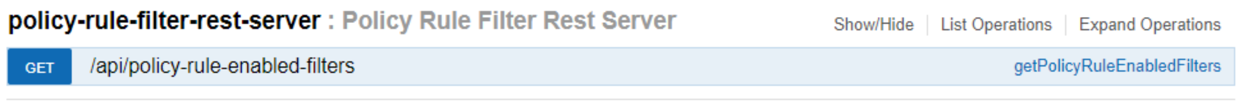
GET	/api/notifications	findNotifications
GET	/api/notifications/{notificationId}	findNotification
GET	/api/users/{userId}/notification-counts	findUserNotificationsCount
PUT	/api/users/{userId}/notification-states	updateAllNotificationState
GET	/api/users/{userId}/notifications	findUserNotifications
GET	/api/users/{userId}/notifications/{notificationId}	findUserNotification
PUT	/api/users/{userId}/notifications/{notificationId}	updateUserNotification



notification-subscription-rest-server : Notification Subscription Rest Server

Show/Hide | List Operations | Expand Operations

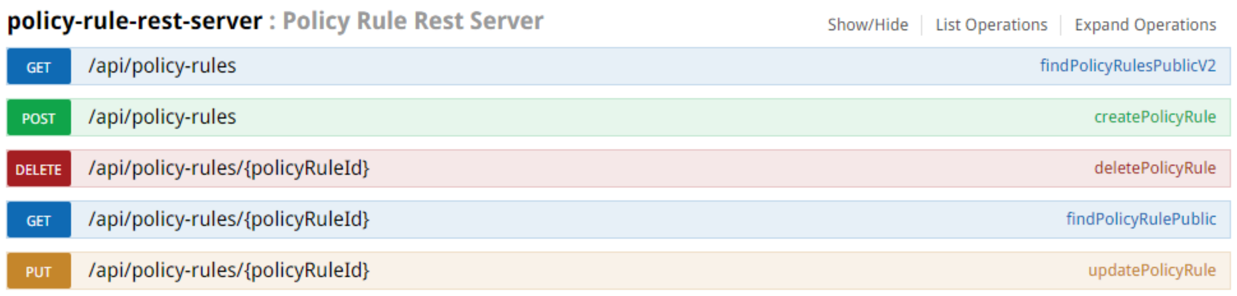
GET	/api/users/{userId}/notification-subscriptions	findNotificationSubscriptions
POST	/api/users/{userId}/notification-subscriptions	createNotificationSubscription
DELETE	/api/users/{userId}/notification-subscriptions/{subscriptionId}	deleteNotificationSubscription
GET	/api/users/{userId}/notification-subscriptions/{subscriptionId}	findNotificationSubscription



policy-rule-filter-rest-server : Policy Rule Filter Rest Server

Show/Hide | List Operations | Expand Operations

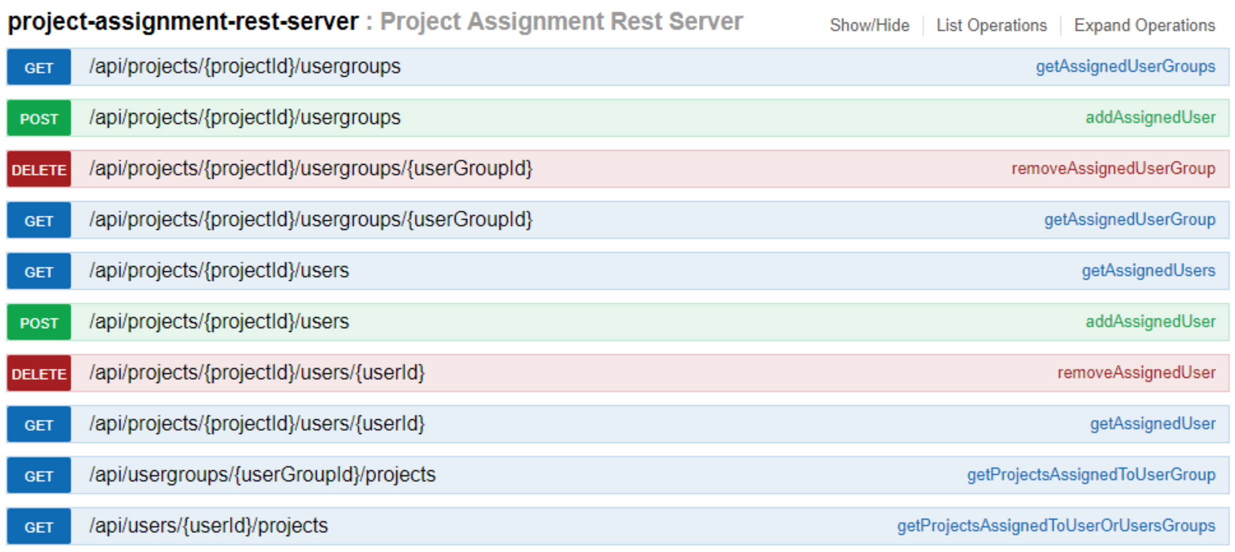
GET	/api/policy-rule-enabled-filters	getPolicyRuleEnabledFilters



policy-rule-rest-server : Policy Rule Rest Server

Show/Hide | List Operations | Expand Operations

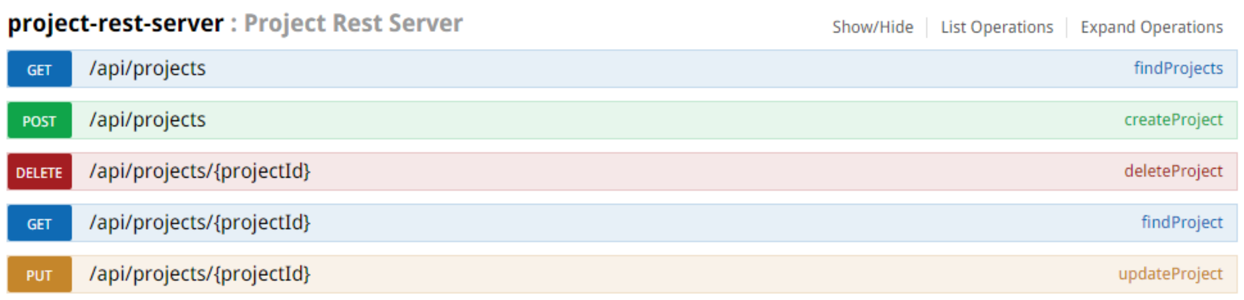
GET	/api/policy-rules	findPolicyRulesPublicV2
POST	/api/policy-rules	createPolicyRule
DELETE	/api/policy-rules/{policyRuleId}	deletePolicyRule
GET	/api/policy-rules/{policyRuleId}	findPolicyRulePublic
PUT	/api/policy-rules/{policyRuleId}	updatePolicyRule



project-assignment-rest-server : Project Assignment Rest Server

Show/Hide | List Operations | Expand Operations

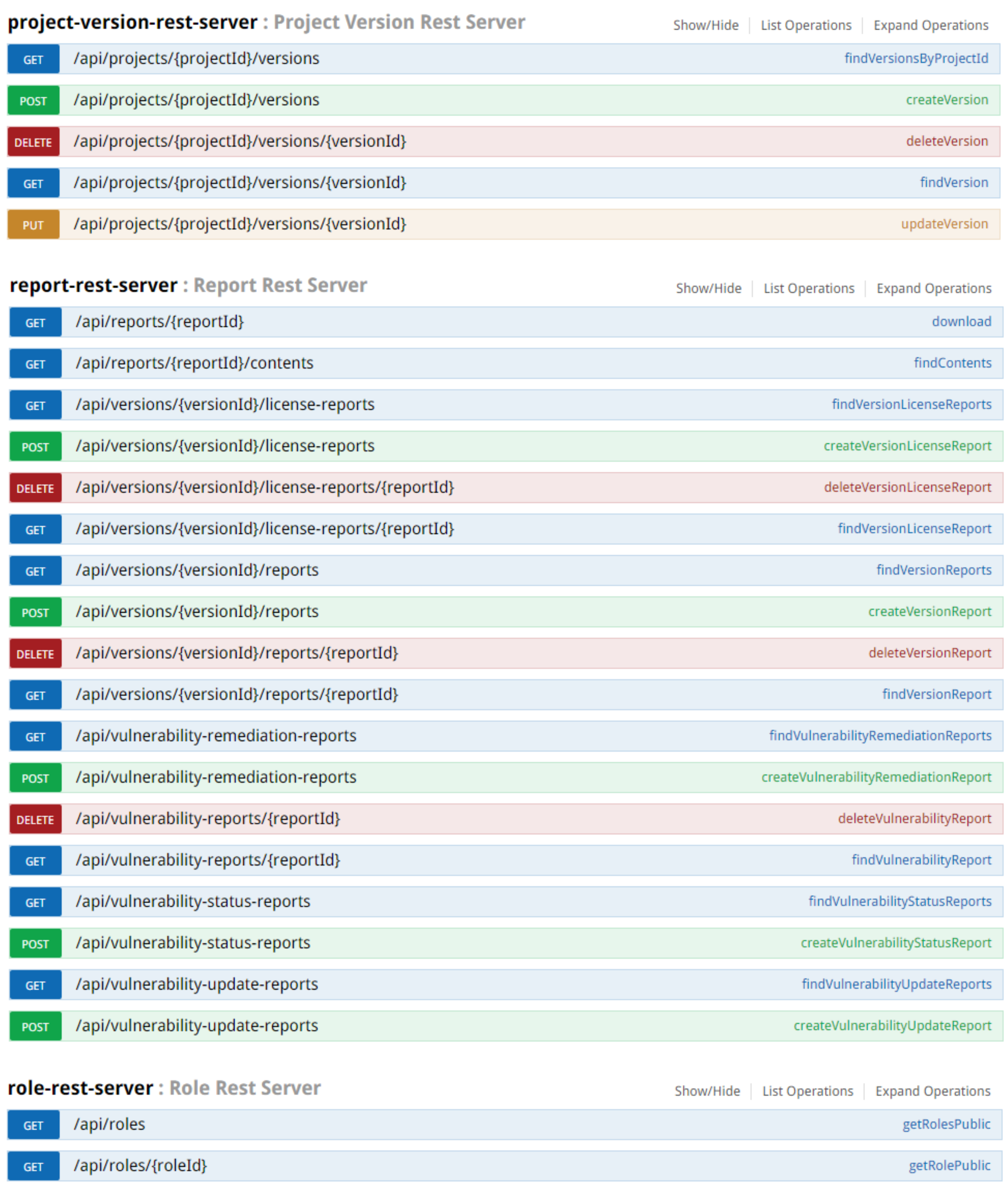
GET	/api/projects/{projectId}/usergroups	getAssignedUserGroups
POST	/api/projects/{projectId}/usergroups	addAssignedUser
DELETE	/api/projects/{projectId}/usergroups/{userGroupId}	removeAssignedUserGroup
GET	/api/projects/{projectId}/usergroups/{userGroupId}	getAssignedUserGroup
GET	/api/projects/{projectId}/users	getAssignedUsers
POST	/api/projects/{projectId}/users	addAssignedUser
DELETE	/api/projects/{projectId}/users/{userId}	removeAssignedUser
GET	/api/projects/{projectId}/users/{userId}	getAssignedUser
GET	/api/usergroups/{userGroupId}/projects	getProjectsAssignedToUserGroup
GET	/api/users/{userId}/projects	getProjectsAssignedToUserOrUsersGroups



project-rest-server : Project Rest Server

Show/Hide | List Operations | Expand Operations

GET	/api/projects	findProjects
POST	/api/projects	createProject
DELETE	/api/projects/{projectId}	deleteProject
GET	/api/projects/{projectId}	findProject
PUT	/api/projects/{projectId}	updateProject



project-version-rest-server : Project Version Rest Server

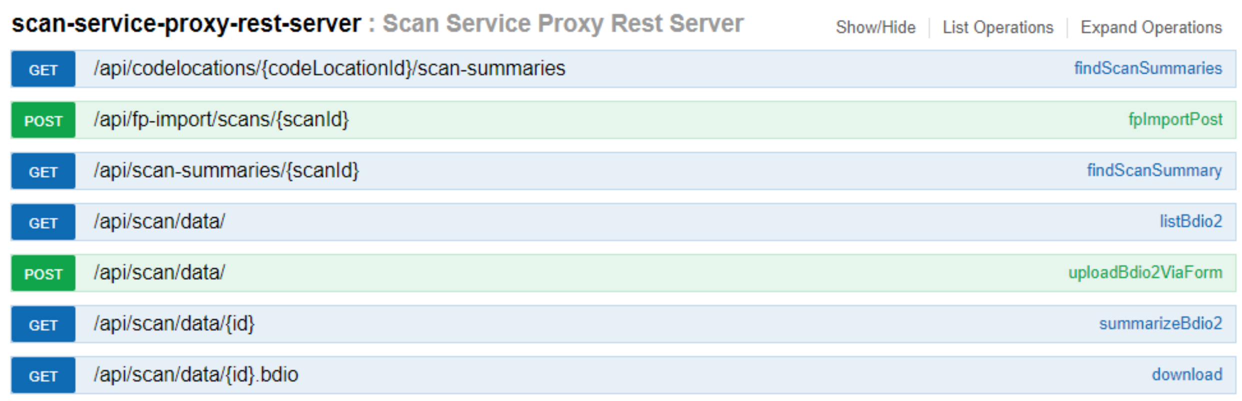
GET	/api/projects/{projectId}/versions	findVersionsByProjectId
POST	/api/projects/{projectId}/versions	createVersion
DELETE	/api/projects/{projectId}/versions/{versionId}	deleteVersion
GET	/api/projects/{projectId}/versions/{versionId}	findVersion
PUT	/api/projects/{projectId}/versions/{versionId}	updateVersion

report-rest-server : Report Rest Server

GET	/api/reports/{reportId}	download
GET	/api/reports/{reportId}/contents	findContents
GET	/api/versions/{versionId}/license-reports	findVersionLicenseReports
POST	/api/versions/{versionId}/license-reports	createVersionLicenseReport
DELETE	/api/versions/{versionId}/license-reports/{reportId}	deleteVersionLicenseReport
GET	/api/versions/{versionId}/license-reports/{reportId}	findVersionLicenseReport
GET	/api/versions/{versionId}/reports	findVersionReports
POST	/api/versions/{versionId}/reports	createVersionReport
DELETE	/api/versions/{versionId}/reports/{reportId}	deleteVersionReport
GET	/api/versions/{versionId}/reports/{reportId}	findVersionReport
GET	/api/vulnerability-remediation-reports	findVulnerabilityRemediationReports
POST	/api/vulnerability-remediation-reports	createVulnerabilityRemediationReport
DELETE	/api/vulnerability-reports/{reportId}	deleteVulnerabilityReport
GET	/api/vulnerability-reports/{reportId}	findVulnerabilityReport
GET	/api/vulnerability-status-reports	findVulnerabilityStatusReports
POST	/api/vulnerability-status-reports	createVulnerabilityStatusReport
GET	/api/vulnerability-update-reports	findVulnerabilityUpdateReports
POST	/api/vulnerability-update-reports	createVulnerabilityUpdateReport

role-rest-server : Role Rest Server

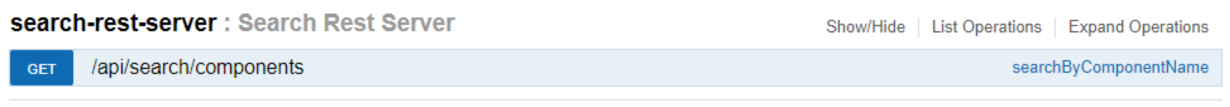
GET	/api/roles	getRolesPublic
GET	/api/roles/{roleId}	getRolePublic



scan-service-proxy-rest-server : Scan Service Proxy Rest Server

Show/Hide | List Operations | Expand Operations

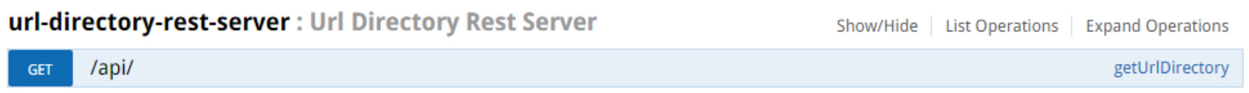
GET	/api/codelocations/{codeLocationId}/scan-summaries	findScanSummaries
POST	/api/fp-import/scans/{scanId}	fpImportPost
GET	/api/scan-summaries/{scanId}	findScanSummary
GET	/api/scan/data/	listBdio2
POST	/api/scan/data/	uploadBdio2ViaForm
GET	/api/scan/data/{id}	summarizeBdio2
GET	/api/scan/data/{id}.bdio	download



search-rest-server : Search Rest Server

Show/Hide | List Operations | Expand Operations

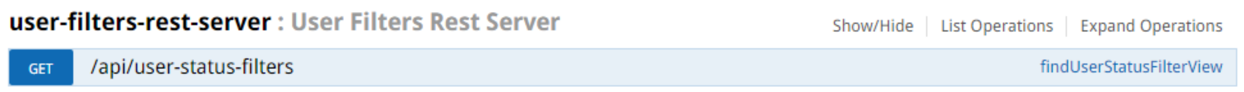
GET	/api/search/components	searchByComponentName



url-directory-rest-server : Url Directory Rest Server

Show/Hide | List Operations | Expand Operations

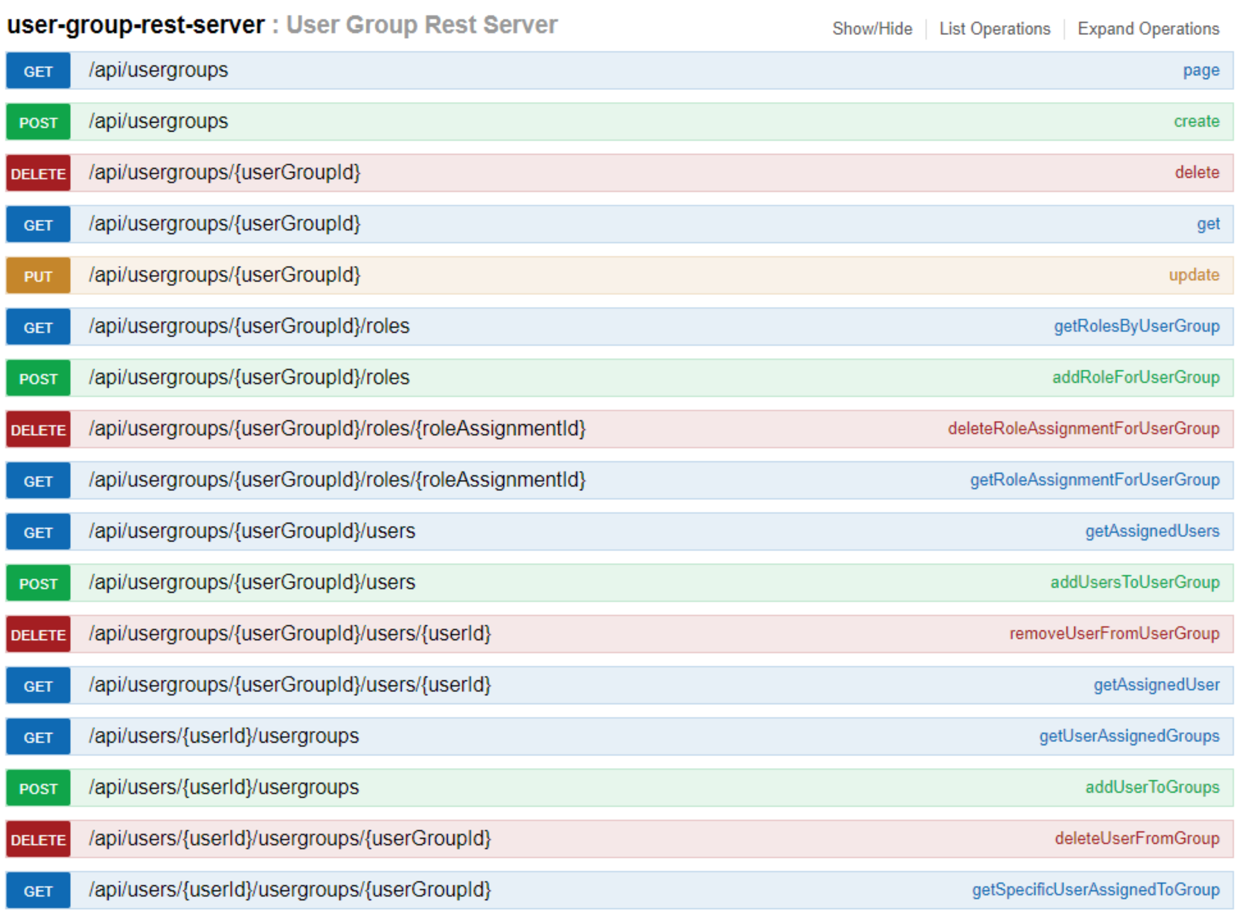
GET	/api/	getUrlDirectory



user-filters-rest-server : User Filters Rest Server

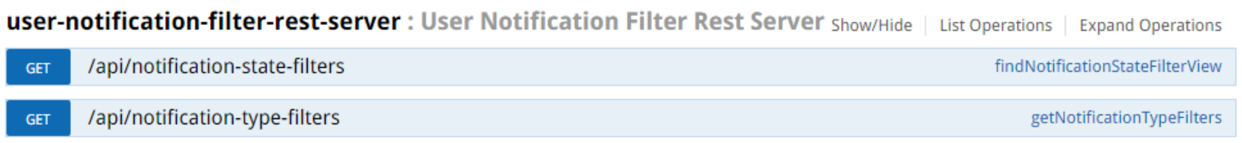
Show/Hide | List Operations | Expand Operations

GET	/api/user-status-filters	findUserStatusFilterView



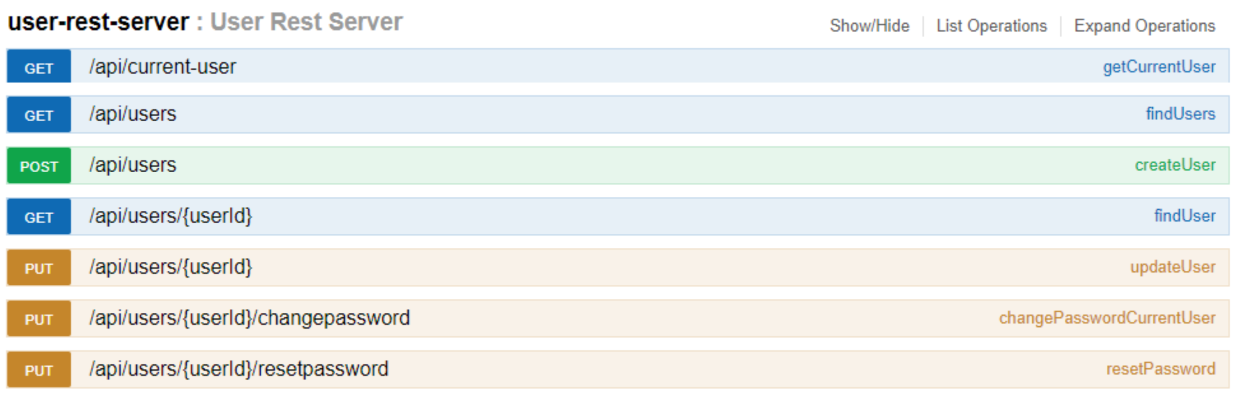
user-group-rest-server : User Group Rest Server   Show/Hide | List Operations | Expand Operations

GET	/api/usergroups	page
POST	/api/usergroups	create
DELETE	/api/usergroups/{userGroupId}	delete
GET	/api/usergroups/{userGroupId}	get
PUT	/api/usergroups/{userGroupId}	update
GET	/api/usergroups/{userGroupId}/roles	getRolesByUserGroup
POST	/api/usergroups/{userGroupId}/roles	addRoleForUserGroup
DELETE	/api/usergroups/{userGroupId}/roles/{roleAssignmentId}	deleteRoleAssignmentForUserGroup
GET	/api/usergroups/{userGroupId}/roles/{roleAssignmentId}	getRoleAssignmentForUserGroup
GET	/api/usergroups/{userGroupId}/users	getAssignedUsers
POST	/api/usergroups/{userGroupId}/users	addUsersToUserGroup
DELETE	/api/usergroups/{userGroupId}/users/{userId}	removeUserFromUserGroup
GET	/api/usergroups/{userGroupId}/users/{userId}	getAssignedUser
GET	/api/users/{userId}/usergroups	getUserAssignedGroups
POST	/api/users/{userId}/usergroups	addUserToGroups
DELETE	/api/users/{userId}/usergroups/{userGroupId}	deleteUserFromGroup
GET	/api/users/{userId}/usergroups/{userGroupId}	getSpecificUserAssignedToGroup



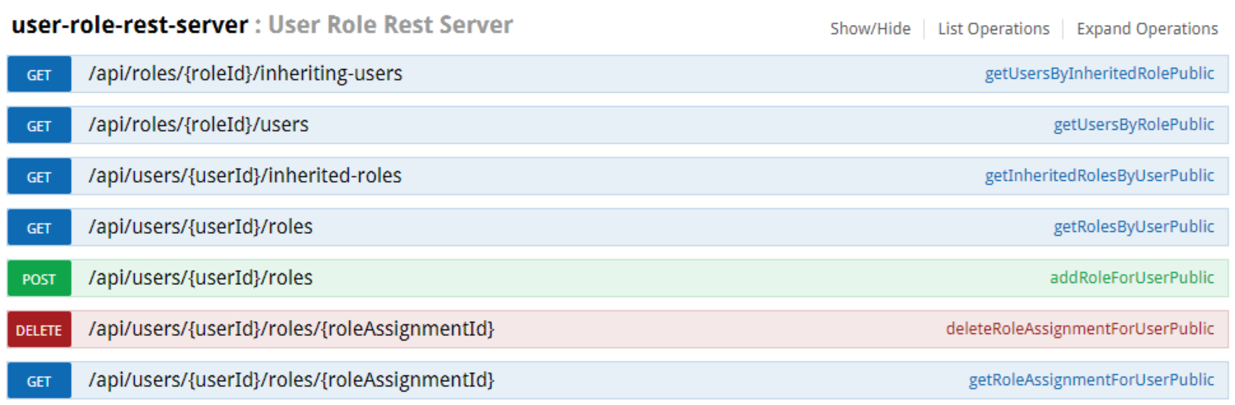
user-notification-filter-rest-server : User Notification Filter Rest Server   Show/Hide | List Operations | Expand Operations

GET	/api/notification-state-filters	findNotificationStateFilterView
GET	/api/notification-type-filters	getNotificationTypeFilters



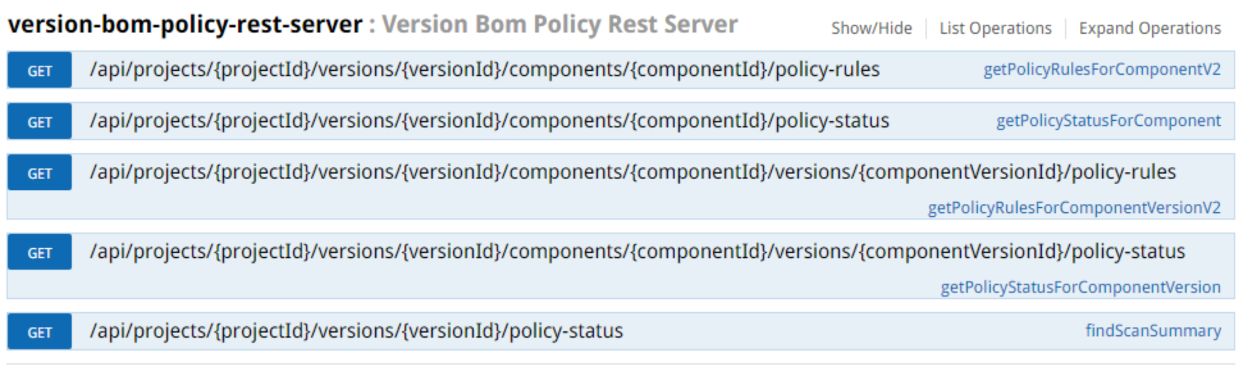
user-rest-server : User Rest Server   Show/Hide | List Operations | Expand Operations

GET	/api/current-user	getCurrentUser
GET	/api/users	findUsers
POST	/api/users	createUser
GET	/api/users/{userId}	findUser
PUT	/api/users/{userId}	updateUser
PUT	/api/users/{userId}/changepassword	changePasswordCurrentUser
PUT	/api/users/{userId}/resetpassword	resetPassword



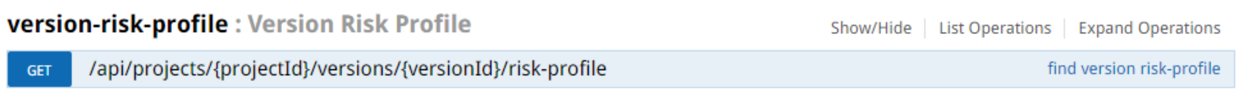
user-role-rest-server : User Role Rest Server

GET	/api/roles/{roleId}/inheriting-users	getUsersByInheritedRolePublic
GET	/api/roles/{roleId}/users	getUsersByRolePublic
GET	/api/users/{userId}/inherited-roles	getInheritedRolesByUserPublic
GET	/api/users/{userId}/roles	getRolesByUserPublic
POST	/api/users/{userId}/roles	addRoleForUserPublic
DELETE	/api/users/{userId}/roles/{roleAssignmentId}	deleteRoleAssignmentForUserPublic
GET	/api/users/{userId}/roles/{roleAssignmentId}	getRoleAssignmentForUserPublic



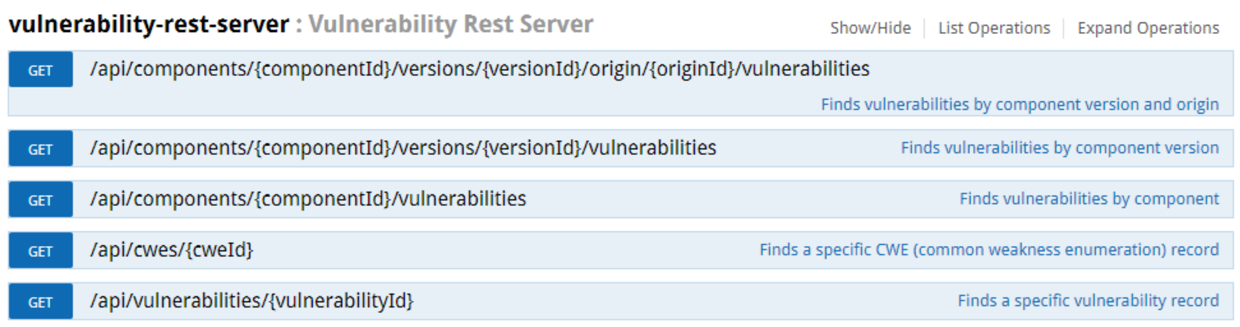
version-bom-policy-rest-server : Version Bom Policy Rest Server

GET	/api/projects/{projectId}/versions/{versionId}/components/{componentId}/policy-rules	getPolicyRulesForComponentV2
GET	/api/projects/{projectId}/versions/{versionId}/components/{componentId}/policy-status	getPolicyStatusForComponent
GET	/api/projects/{projectId}/versions/{versionId}/components/{componentId}/versions/{componentVersionId}/policy-rules	getPolicyRulesForComponentVersionV2
GET	/api/projects/{projectId}/versions/{versionId}/components/{componentId}/versions/{componentVersionId}/policy-status	getPolicyStatusForComponentVersion
GET	/api/projects/{projectId}/versions/{versionId}/policy-status	findScanSummary



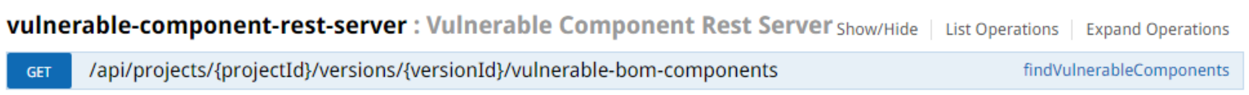
version-risk-profile : Version Risk Profile

GET	/api/projects/{projectId}/versions/{versionId}/risk-profile	find version risk-profile



vulnerability-rest-server : Vulnerability Rest Server

GET	/api/components/{componentId}/versions/{versionId}/origin/{originId}/vulnerabilities	Finds vulnerabilities by component version and origin
GET	/api/components/{componentId}/versions/{versionId}/vulnerabilities	Finds vulnerabilities by component version
GET	/api/components/{componentId}/vulnerabilities	Finds vulnerabilities by component
GET	/api/cwes/{cweId}	Finds a specific CWE (common weakness enumeration) record
GET	/api/vulnerabilities/{vulnerabilityId}	Finds a specific vulnerability record



vulnerable-component-rest-server : Vulnerable Component Rest Server

GET	/api/projects/{projectId}/versions/{versionId}/vulnerable-bom-components	findVulnerableComponents