## Table of Contents

[Add Endpoints](#_s6xg1ypx6kqs)

[add/NewEngagement](#_cnk6omd6sdsj)

[add/NewTestPlan](#_3a1tbj5jct45)

[add/NewTestCase](#_awdkqol3i1o9)

[add/NewTest](#_cietzu49y5q7)

[add/NewResult](#_f2paeal3rnb6)

[addBOMDevices](#_b1w0xsk0ok5g)

[addLibraryDevice](#_h1xbx2hs5v66)

[Clone Endpoints](#_v1wyi3x9s4ny)

[cloneTestPlan](#_4lejnpyviv5v)

[cloneTestCase](#_z1crf4cekcuf)

[cloneTest](#_b4yjlrvto6yn)

[Get Endpoints](#_4fp4cts24g9y)

[getActiveEngagements](#_ywsj45waloy4)

[getEngagementDetails](#_8ozxfi4o2sc6)

[getEngagement](#_dw54gmxpgnla)

[getTestPlansByEngagementId](#_ehuvd5yth441)

[getTestPlans](#_7vu94jjdizwm)

[getTestCasesByTestPlan](#_t5e6sw4et5j6)

[getResults](#_ar3a4ul9v7j6)

[getLibraryTestPlans](#_o1e3pdc5xboo)

[getLibraryTestCases](#_qodfant46rgo)

[getLibraryTests](#_sqn6vdq0f085)

[getAllDevices](#_w88q5lwi2w1x)

[Edit Endpoints](#_kgda82pqbnmu)

[edit/Engagement](#_tzcgv7qvw9ne)

[edit/TestPlan](#_p63a0c2c0qa0)

[edit/TestCase](#_ktbffrlevkze)

[edit/Test](#_ds4bvi4d707n)

[edit/Result](#_pc2xxzo63k98)

[edit/ActivateTestPlan](#_gzfyh0daxqs8)

[editDeviceInBOM](#_fhkhum5kvv3w)

[Delete Endpoints](#_ib77q0mq29pz)

[deleteEngagement](#_srtsdajvv7r5)

[deleteTestPlan](#_vm6mq0mm2p9f)

[deleteTestCase](#_hbcj1m60anlq)

[deleteTest](#_4ekmczx656bf)

[deleteTestCaseBOM](#_rg92q46oubaz)

[deleteResult](#_87v67lfc1vt3)

## 

## Add Endpoints

#### add/NewEngagement

* Method: POST Request
* Inputs: Json that contains proper Engagement fields according to engagementSchema.
* Outputs:
  + 200: Successful Request, returns results of the mongo Query
  + 400: Query Error, returns error type / message/ results of the mongo Query
  + 405: Not POST request
  + 422: Validation Error, returns error type / message / summary of errors
  + 500: Connection Error / Failed to connect to DB, returns error type / message
    - Also catches any other errors that may occur, such as in the code logic
* Description: Endpoint creates a New Engagement.

#### add/NewTestPlan

* Method: POST Request
* Inputs: Json that contains proper TestPlan fields according to testPlanSchema.
* Outputs:
  + 200: Successful Request, returns results of the mongo Query
  + 400: Query Error, returns error type / message/ results of the mongo Query
  + 405: Not POST request
  + 422: Validation Error, returns error type / message / summary of errors
  + 500: Connection Error / Failed to connect to DB, returns error type / message
    - Also catches any other errors that may occur, such as in the code logic
* Description: Endpoint creates a New TestPlan.

#### add/NewTestCase

* Method: POST Request
* Inputs: Json that contains proper TestCase fields according to testCaseSchema.
* Outputs:
  + 200: Successful Request, returns results of the mongo Queries
    - Contains two queries (queryResult and testPlanUpdateResult): the first inserts the TestCase, the second updates the parent TestPlan’s child references
  + 400: Query Error, returns error type / message/ results of the mongo Query
    - May fail for either queryResult or testPlanUpdateResult
  + 405: Not POST request
  + 422: Validation Error, returns error type / message / summary of errors
  + 500: Connection Error / Failed to connect to DB, returns error type / message
    - Also catches any other errors that may occur, such as in the code logic
* Description: Endpoint creates a New TestCase.

#### add/NewTest

* Method: POST Request
* Inputs: Json that contains proper Test fields according to testSchema.
* Outputs:
  + 200: Successful Request, returns results of the mongo Queries
    - Contains two queries (queryResult and testCaseUpdateResult): the first inserts the TestCase, the second updates the parent TestPlan’s child references
  + 400: Query Error, returns error type / message/ results of the mongo Query
    - May fail for either queryResult or testCaseUpdateResult
  + 405: Not POST request
  + 422: Validation Error, returns error type / message / summary of errors
  + 500: Connection Error / Failed to connect to DB, returns error type / message
    - Also catches any other errors that may occur, such as in the code logic
* Description: Endpoint creates a New Test.

#### add/NewResult

* Method: POST Request
* Inputs: Json that contains proper Test fields according to testSchema.
* Outputs:
  + 200: Successful Request, returns results of the mongo Queries
    - Contains two queries (queryResult and testCaseUpdateResult): the first inserts the TestCase, the second updates the parent TestPlan’s child references
  + 400: Query Error, returns error type / message/ results of the mongo Query
    - May fail for either queryResult or testCaseUpdateResult
  + 405: Not POST request
  + 422: Validation Error, returns error type / message / summary of errors
  + 500: Connection Error / Failed to connect to DB, returns error type / message
    - Also catches any other errors that may occur, such as in the code logic
* Description: Endpoint creates a New Result.

#### addBOMDevices

* Method: POST request
* Inputs:
  + devices field containing a list of proper bomDevices with fields according to bomDevice Schema.
  + testCaseId field
  + testPlanId field
* Outputs:
  + 200: Successful Request
  + 405: Not POST request
  + 500: Returns the message of any error that occurred
* Description: Given a list of devices, and the Id’s of the parent TestCase and TestPlan, it adds the list of devices to the TestCase BOM and the TestPlan summaryBOM.

#### addLibraryDevice

* Method: POST request
* Inputs: Json that contains proper libraryDevice fields according to libraryDeviceSchema
* Outputs:
  + 200: Successful Request
  + 405: Not POST request
  + 500: Returns the message of any error that occurred
* Description: Adds a device to the library

## 

## Clone Endpoints

#### cloneTestPlan

* Method: POST request
* Inputs: Json that contains proper TestPlan fields according to testPlanSchema
* Outputs:
  + 200: Success
  + 405: Not POST request
  + 500: Failure
* Description: clones a test plan from test plan library (including the test cases, tests, and BOMs)

#### cloneTestCase

* Method: POST request
* Inputs: Json that contains proper TestCase fields according to testCaseSchema
* Outputs:
  + 200: Success
  + 405: Not POST request
  + 500: Failure
* Description: clones a test case from test case library (including the tests and BOM)

#### cloneTest

* Method: POST request
* Inputs: Json that contains proper Test fields according to testSchema
* Outputs:
  + 200: Success
  + 405: Not POST request
  + 500: Failure
* Description: clones a test from test library

## 

## Get Endpoints

#### getActiveEngagements

* Method: GET Request
* Inputs: /
* Outputs: List of engagements
* Description: get all the engagement with an active status (status code greater than 13)

#### getEngagementDetails

* Method: GET Request
* Inputs: Engagement id
* Outputs: A json file that contains all the Engagement fields, and all details of its nested layers (test plans, test cases, and tests)
* Description: get all the engagement details by id (use case: export the Engagement as json)

#### getEngagement

* Method: GET Request
* Inputs: Engagement id
* Outputs: List of engagements that has the engagement id
* Description: get engagement by id, though a list is returned, there should be either 0 or 1 items in this list

#### getTestPlansByEngagementId

* Method: GET Request
* Inputs: engagement id
* Outputs: List of test plans
* Description: get all the test plans belong to the specific engagement

#### getTestPlans

* Method: GET request
* Inputs: engagementID
* Outputs: List of test plans
* Description: returns an array with all the archived test plans associated with an engagement

#### getTestCasesByTestPlan

* Method: GET Request
* Inputs: test plan id
* Outputs: List of test cases
* Description: get all the test cases belong to the specific test plan

#### getResults

* Method: GET request
* Inputs: test id
* Outputs: list of results
* Description: get all the results of the specific test

#### getLibraryTestPlans

* Method: GET Request
* Inputs: /
* Outputs: A list of test plans
* Description: get all test plans from the test plan library

#### getLibraryTestCases

* Method: GET Request
* Inputs: /
* Outputs: A list containing all the test cases in the test case library
* Description: get all test cases from the library of test cases

#### getLibraryTests

* Method: GET Request
* Inputs: /
* Outputs: A list of tests
* Description: get all tests from the test library

#### getAllDevices

* Method: GET Request
* Inputs: /
* Outputs: List of devices
* Description: get all the library devices

## 

## Edit Endpoints

#### edit/Engagement

* Method: PUT request
* Inputs: Json that contains proper Engagement fields according to engagementSchema
  + 200: Successful Request, returns results of the mongo Query
  + 400: Query Error, returns error type / message/ results of the mongo Query
  + 405: Not PUT request
  + 422: Validation Error, returns error type / message / summary of errors
  + 500: Connection Error / Failed to connect to DB, returns error type / message
    - Also catches any other errors that may occur, such as in the code logic
* Description: updates an existing engagement with the provided fields

#### edit/TestPlan

* Method: PUT request
* Inputs: Json that contains proper TestPlan fields according to testPlanSchema.
* Outputs:
  + 200: Successful Request, returns results of the mongo Query
  + 400: Query Error, returns error type / message/ results of the mongo Query
  + 405: Not PUT request
  + 422: Validation Error, returns error type / message / summary of errors
  + 500: Connection Error / Failed to connect to DB, returns error type / message
    - Also catches any other errors that may occur, such as in the code logic
* Description: updates an existing TestPlan with the provided fields

#### edit/TestCase

* Method: PUT request
* Inputs: Json that contains proper TestCase fields according to testCaseSchema.
* Outputs:
  + 200: Successful Request, returns results of the mongo Query
  + 400: Query Error, returns error type / message/ results of the mongo Query
  + 405: Not PUT request
  + 422: Validation Error, returns error type / message / summary of errors
  + 500: Connection Error / Failed to connect to DB, returns error type / message
    - Also catches any other errors that may occur, such as in the code logic
* Description: updates an existing TestCase with the provided fields

#### edit/Test

* Method: PUT request
* Inputs: Json that contains proper Test fields according to testSchema.
* Outputs:
  + 200: Successful Request, returns results of the mongo Query
  + 400: Query Error, returns error type / message/ results of the mongo Query
  + 405: Not PUT request
  + 422: Validation Error, returns error type / message / summary of errors
  + 500: Connection Error / Failed to connect to DB, returns error type / message
    - Also catches any other errors that may occur, such as in the code logic
* Description: updates an existing Test with the provided fields

#### edit/Result

* Method: PUT request
* Inputs: Json that contains proper Result fields according to resultSchema.
* Outputs:
  + 200: Successful Request, returns results of the mongo Query
  + 400: Query Error, returns error type / message/ results of the mongo Query
  + 405: Not PUT request
  + 422: Validation Error, returns error type / message / summary of errors
  + 500: Connection Error / Failed to connect to DB, returns error type / message
    - Also catches any other errors that may occur, such as in the code logic
* Description: updates an existing Result with the provided fields

#### edit/ActivateTestPlan

* Method: PUT request
* Inputs:
  + engagementId
  + testPlanId
* Outputs:
  + 200: Successful Request, returns results of the mongo Query
  + 400: Query Error, returns error type / message/ results of the mongo Query
  + 405: Not PUT request
  + 500: Connection Error / Failed to connect to DB, returns error type / message
    - Also catches any other errors that may occur, such as in the code logic
* Description: Makes the specified test plan active for the corresponding parent engagement

#### editDeviceInBOM

* Method: PUT request
* Inputs:
  + testCaseId
  + testPlanId
  + devices (a list of BOM devices, though there should be exactly 1 element)
* Outputs:
  + 200: Successful Request, returns results of the mongo Query
  + 405: Not PUT request
  + 422: Validation Error, returns error type / message / summary of errors
  + 500: Connection Error / Failed to connect to DB, returns error type / message
* Description: Updates the input device entry in BOM of the corresponding test case; automatically updates the summaryBOM of the corresponding test plan

## Delete Endpoints

#### deleteEngagement

* Method: POST request
* Inputs: engagement id
* Outputs:
  + 200: Success
  + 405: Not POST request
  + 500: Failure
* Description: Deletes an engagement

#### deleteTestPlan

* Method: POST request
* Inputs: \_id (test plan id), parentEngagementId
* Outputs:
  + 200: Success
  + 405: Not POST request
  + 500: Failure
* Description: Deletes a test plan, along with all the nested layers

#### deleteTestCase

* Method: POST request
* Inputs: \_id (test case id), parentTestPlanId
* Outputs:
  + 200: Success
  + 405: Not POST request
  + 500: Failure
* Description: Deletes a test plan, along with all the nested layers

#### deleteTest

* Method: POST request
* Inputs: \_id (test id), parentTestCaseId
* Outputs:
  + 200: Success
  + 405: Not POST request
  + 500: Failure
* Description: Deletes a test, along with all the nested layers

#### deleteTestCaseBOM

* Method: POST request
* Inputs: \_id (id of the BOM entry), parentTestCaseId
* Outputs:
  + 200: Success
  + 405: Not POST request
  + 500: Failure
* Description: Deletes a device entry in the BOM of the parent test case; automatically updates the summaryBOM of the corresponding test plan

#### deleteResult

* Method: POST request
* Inputs: \_id (result id), parentTestId
* Outputs:
  + 200: Success
  + 405: Not POST request
  + 500: Failure
* Description: Deletes a result