CHARTER

-----------------------------------------------

Identify capabilities and areas of potential instability of the “rest api project list manager”

Build

-----------------------------------------------

Java -jar runTOdoManagerRestAPI-1.5.5.jar

Area

-----------------------------------------------

Main functions and capabilities

Environment

-----------------------------------------------

Windows 11 Pro Version 23H2

Screen resolution: 1920 x 1080

START

-----------------------------------------------

4:54 PM 09/28/2024

Other Data

-----------------------------------------------

TESTER

-----------------------------------------------

Andrew Chirita

[andrew.chirita@mail.mcgill.ca](mailto:andrew.chirita@mail.mcgill.ca)

261051967

#DURATION

45 minutes

4:54 PM (1): GET /projects request appears functional. Each instance of a project along with related tasks are displayed.

4:56 PM (2): HEAD /projects request with one project created appears functional. No data returned from the json response except from header data.

4:58 PM (3): POST /projects request without request parameters successfully creates a project with default values.

5:00 PM (4): POST /projects request with valid request parameters and an id in the input expectedly raises an error.

5:02 PM (5): POST /projects request with invalid request parameters increments the id counter without a project being created.

5:04 PM (6): POST /projects request with valid request parameters correctly creates a new project.

5:06 PM (7): GET /projects/1 request appears functional. It returns the project with the corresponding id = 1.

5:08 PM (8): GET /projects/7 request expectedly raises an error and does not return a project when there is no project with an assigned id of 7.

5:10 PM (9): HEAD /projects/1 request appears functional. It returns the header information for a project with the corresponding id = 1.

5:12 PM (10): HEAD /projects/7 request expectedly raises an error and does not return response header when there is no project with an assigned id of 7.

5:14 PM (11): POST /projects/1 request with valid body values correctly modifies the project with corresponding id = 1.

5:16 PM (12): POST /projects/7 request expectedly raises an error and does not return modify a project when there is no project with an assigned id of 7.

5:18 PM (13): PUT /projects/1 request with valid body values correctly modifies the project with corresponding id = 1.

5:20 PM (14): PUT /projects/7 request expectedly raises an error and does not modify a project when there is no project with an assigned id of 7.

5:22 PM (15): DELETE /projects/1 request correctly deletes the project with corresponding id = 1.

5:24 PM (16): DELETE /projects/7 request expectedly raises an error and does not return modify a project when there is no project with an assigned id of 7.

5:26 PM (17): GET /projects/2/tasks correctly returns the list of tasks for the project with corresponding id = 2.

5:28 PM (18): GET /projects/7/tasks incorrectly returns a response with all todos although the project with corresponding id = 7 doesn’t exist.

5:30 PM (19): HEAD /projects/2/tasks correctly returns the header information for the project with corresponding id = 2.

5:32 PM (20): HEAD /projects/7/tasks returns the header information for the project tasks with corresponding id = 7, although the project with id = 7 does not exist.

5:34 PM (21): POST /projects/2/tasks correctly creates a new todo and creates a task relationship whenever a title value is defined in the request body, but not an id.

5:36 PM (22): POST /projects/2/tasks correctly creates a task relationship by assigning an existing todo whenever an id value is defined in the request body.

5:38 PM (23): DELETE /projects/2/tasks/1 correctly deletes the todo with id = 1 associated to the project with id = 2.

5:39 PM (24): DELETE /projects/7/tasks/7 correctly raises an error whenever a project with the corresponding id = 7 doesn’t exist.

5:39 PM (25): DELETE /projects/2/tasks/5 correctly raises an error whenever a task with corresponding id = 5 doesn’t exist.

END TIME:

--------------------------------------------------

5:39 PM 09/28/2024

SUMMARY OF SESSION FINDINGS

-----------------------------------------------

The GET operations perform as expected aside from retrieving tasks associated with a project. Whenever the id doesn’t correspond to an existing project, the GET request still obtains an empty response.

The HEAD operations correctly retrieve header information regarding projects. Again, however, when retrieving the header for projects tasks with an non-existent id, the request still obtains a response.

The POST operations mostly behave correctly. An incorrect request body results in the id counter being wrongfully incremented. When the request body has valid inputs, the operation behaves correctly. When a task needs to be created based on a POST operation, the operation correctly creates a new todo before linking it to the project, assuring that the project operations are interoperable with the todo operations. Whenever an existing todo id is provided in the body, the POST operation assures that the existing todo is linked to the project instead of creating a new todo.

The PUT operations behave similarly to the POST operations, and do not have any faulty behaviour based on the session findings.

The DELETE operations behave as expected. Whenever an existing id is given, the project with that id gets correctly deleted. Whenever a project with the given id does not exist, the operation raises an error.

LIST OF CONCERNS

---------------------------------------------------

The project id counter is incremented despite there being a wrongful POST operation

HEAD and GET operations for tasks based on non existent project ids still return good responses

Potential api rate limits

The data is volatile, therefore any disconnects could result in a loss of data

NEW TESTING IDEAS:

Test every operation with invalid data types in the request body

Test every operation with existing ids and non existent ids

Compare outputs between PUT and POST operations

----------------------------------------------------

FILES REFERENCED OR CREATED DURING THE SESSION

-----------------------------------------------

runTodoManagerRestAPI-1.5.5.jar (API build file)

POSTMAN\_PROJECTS (Screenshots of the script)