**CHARTER**

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

Identify documented and undocumented “rest api todo list manager” capabilities.

For each capability create a script or small program to demonstrate the capability.

Exercise each capability identified with data typical to the intended use of the application.

**BUILD**

runTodoManagerRestAPI-1.5.5.jar

**AREA**

Test “project” feature

**ENVIRONMENT**

Windows 11

**START**

5 :00 PM 02/12/2025

**DURATION**

45 mins

**TESTER**

Neeshal Imrit 261068305 [neeshal.imrit@mail.mcgill.ca](mailto:neeshal.imrit@mail.mcgill.ca)

Dmytro Martyniuk 261118276 neeshal.imrit@mail.mcgill.ca

**SESSION NOTES**

5:00 PM – 12/02/2025: Set request headers to accept JSON data.

5:01 PM: Sent a GET request to the /projects endpoint to retrieve all projects. Response returns the full list of projects as expected.

5:06 PM: Tested the HEAD request for /projects. Returns status code 200 as expected.

5:10 PM: Created a new project with XML payload (POST /projects), specifying a title and description. The project was successfully created: Status code: 200

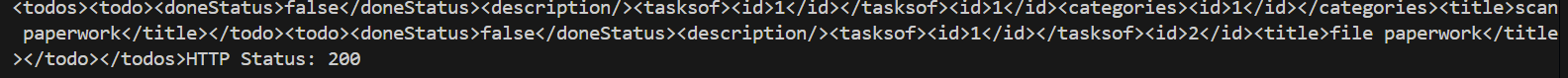
5:12 PM: Fetch exiting project with ID 1 using (Get/projects/1). Returned expected project in Json format.

5:14 PM: Fetching non existing project (Get projects/20). Returns expected error code 404.

5:20 PM: Delete existing project with ID 1 (Delete projects/1). Successfully deleted project with id 1.

5:24 PM: Delete non existing project with ID 20 (Delete projects/1). Returns an appropriate error message as expected.

5:26 PM: Get request to projects endpoint (GET /projects/1/tasks) to return al todo items related to the project id. Returns all todo items related to the specified project id. Works as expected

5:30 PM: Get request to get all todos related to a non-existent project (GET /projects/20/task). Returns another project. ## BUG (see Screenshot)

5:35 PM: Head request for task related to a specific project (HEAD /projects/1/Tasks). Returns expected value

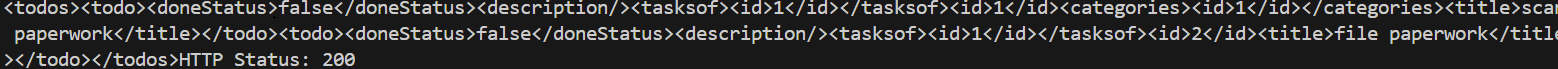
5:37 PM: Add a task to an existing project (POST /projects/:id/tasks) with the task ID in the body. Creates expected relationship.

5:45 PM: Deleting an existing project-category relation (DELETE /projects/:id/categories/:id). Returns status code. Verified with a get request and returned an error indicating that it worked.

**BUGS LOGGED**

1) Endpoint: /projects /:id/tasks, Request Type: GET

Description: Trying to get all the to-dos related to a non-existent project, it should return an error message but it returned the to-dos of another project.



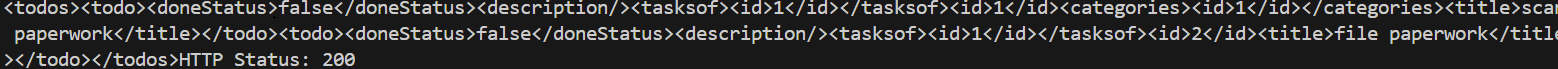
Risk: Data concern. The user should not have access to this data.

**AREAS OF POTENTIAL RISK**

Data Exposure:

Bug #1: Returns data for an invalid request (nonexistent project).

Risk: Users should not have access to data for invalid requests. This could lead to unintended data exposure.



Refer to bug summary: projectsBug1.docx

**CAPABILITIES**

Create, Read, Update, and Delete categories using XML and JSON payloads.

Retrieve headers for projects endpoints.

Retrieve and delete relationships between projects and categories/todos.

Refer to capabilities script: projectJSON.sh and projectXML.sh

**SUMMARY OF SESSION FINDINGS**

* The API supports CRUD operations for projects.
* We can create, retrieve and delete relationships between categories and todos.
* The API returns status 200 when querying tasks related to a non-existent project. It returns the value of other tasks. Users should not have access to that data.

**NEW TESTING IDEAS**

* Verify the relation between projects and tasks/categories
* Verify the status code of each crud operations.