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

Typical Scenarios – main functions and capabilities of relationships between todos, categories and projects

Exploratory testing for API endpoints

Environment

MacOS 26.3

Postman software

START

1:45 PM 13/02/2026

TESTER

Ethan Wu	261117309	ethan.wu2@mail.mcgill.ca
Kenny Nguyen	261120429	kenny.nguyen@mail.mcgill.ca

#DURATION

---

50 minutes

TEST NOTES

-- interoperability of todos and categories

1:45 PM 13/02/2026 [POST /todos/ :id/ categories] Post request to categories of todos endpoint with a specific todo id. Successfully creates a relationship between the specific todo and the category with id specified in the body.

1:47 PM 13/02/2026 [GET /todos/ :id/ categories] Get request to category of todos endpoint with specific todo id. Return the category of the specific todo as expected. Status 200 OK.

1:48 PM 13/02/2026 [POST /todos/ :id/ categories] Post request to categories of todos endpoint with a specific todo id. Since the category id was not specified in the body of the request, status 400 BAD REQUEST.

1:50 PM 13/02/2026 [POST /todos/ :id/ categories] Post request to categories of todos endpoint with an invalid id. Status 404 NOT FOUND as expected.

1:51 PM 13/02/2026 [GET /todos/ :id/ categories] Get request to categories of todos endpoint with an invalid id. This returns the list of categories of todo while it is not supposed to, with status 200 OK. This is a bug.

???bug

1:53 PM 13/02/2026 [GET /todos/ categories] Get request to categories of todos endpoint without todo id.This returns the list of categories of todos with status 200 OK. This is an undocumented API.

1:53 PM 13/02/2026 [HEAD /todos/ :id/ categories] Head request to category of todos endpoint with specific todo id. Return the category of the specific todo as expected. Status 200 OK.

1:55 PM 13/02/2026 [HEAD /todos/ :id/ categories] Head request to categories of todos endpoint with an invalid id. This returns a header while it is not supposed to, with status 200 OK. This is a bug.

???bug

everything in xml doesnt work

1:56 PM 13/02/2026 [POST /todos/ :id/ categories] Post request to categories of todos endpoint with a specific todo id and in XML format. Does not return anything even if the format is the same as the api asks. Status 400 BAD REQUEST.

??bug

1:58 PM 13/02/2026 [GET /todos/ :id/ categories] Get request to category of todos endpoint with specific todo id and in xml format. Does not return anything even if the format is the same as the api asks. Status 400 BAD REQUEST.

1:59 PM 13/02/2026 [HEAD /todos/ :id/ categories] Head request to category of todos endpoint with specific todo id and in xml format. Does not return anything even if the format is the same as the api asks. Status 400 BAD REQUEST.

XML doesnt work

2:00 PM 13/02/2026 [DELETE /todos/ :id/ categories] Delete request to category of todos endpoint with specific todo id. Successfully deletes the correct relationship. Status 200 OK

2:01 PM 13/02/2026 [DELETE /todos/ :id/ categories] Delete request to category of todos endpoint with invalid todo id. Fail to find the id as expected. Status 404 NOT FOUND

2:04 PM 13/02/2026 [POST /categories/ :id/ todos] Post request to todos of categories endpoint with valid category id. Successfully creates a new relationship between the category with specific id and the todo with id specified in the body. Status 201 CREATED.

2:06 PM 13/02/2026 [GET /categories/ :id/ todos] Get request to the todos of categories endpoint with specific category id. Successfully returns the categories of the specific todo as expected. Status 200 OK. Note: it returns the relationships that were created with POST todos -> categories and POST categories -> todos

2:07 PM 13/02/2026 [HEAD /categories/ :id/ todos] Head request to todos of categories endpoint with specific category id. Return the category of the specific todo as expected. Status 200 OK.

2:08PM 13/02/2026 [GET /categories/ todos] Get request to todos of categories endpoint without category id.This returns the list of todos of categories with status 200 OK. This is an undocumented API.

2:10 PM 13/02/2026 [GET /categories/ :id/ todos] Get request to todos of categories endpoint with invalid category id.This returns the list of todos of categories while it is not supposed to, with status 200 OK. This is a bug.

???bug

2:12 PM 13/02/2026 [HEAD /categories/ :id/ todos] Headrequest to todos of categories endpoint with specific category id. This returns a header while it is not supposed to, with status 200 OK. This is a bug.

???bug

2:14 PM 13/02/2026 [DELETE /categories/ :id/ todos] Delete request to todos of categories endpoint with specific category id. Successfully deletes the correct relationship. Status 200 OK.

2:15 PM 13/02/2026 [DELETE /categories/ :id/ todos] Delete request to category of todos endpoint with invalid todo id. Fail to find the id as expected. Status 404 NOT FOUND.

-- interoperability of projects and categories

2:17 PM 13/02/2026 [POST /projects / :id/ categories] Post request to projects of categories endpoint with a specific project id. Successfully creates a relationship between the specific project and the category with id specified in the body.

2:18 PM 13/02/2026 [GET /projects / :id/ categories] Get request to projects of categories endpoint with specific project id. Return the categories of the specific project as expected. Status 200 OK.

2:19 PM 13/02/2026 [POST /projects / :id/ categories] Post request to projects of categories endpoint with a specific project id. Since the category id was not specified in the body of the request, status 400 BAD REQUEST.

2:19 PM 13/02/2026 [POST /projects / :id/ categories] Post request to projects of categories endpoint with an invalid projects id. Status 404 NOT FOUND as expected.

2:20 PM 13/02/2026 [GET /projects/ :id/ categories] Get request to projects of categories endpoint with an invalid projects id. This returns the list of projects of categories while it is not supposed to, with status 200 OK. This is a bug.

???bug

2:22 PM 13/02/2026 [GET /projects/categories] Get request to projects of categories endpoint without projects id.This returns the list of categories of categories with status 200 OK. This is an undocumented API.

2:23 PM 13/02/2026 [HEAD /projects/ :id/ categories] Head request to projects of categories endpoint with specific projects id. Return the header of the categories. Status 200 OK.

2:23 PM 13/02/2026 [HEAD /projects/ :id/ categories] Head request to projects of categories endpoint with an invalid id. This returns a header while it is not supposed to, with status 200 OK. This is a bug.

???bug

2:24 PM 13/02/2026 [POST /projects/ :id/ categories] Post request to categories of projects endpoint with a specific project id and in xml format. Does not return anything even if the format is the same as the api asks. Status 400 BAD REQUEST.

2:26 PM 13/02/2026 [GET /projects/ :id/ categories] Get request to categories of projects endpoint with specific project id and in xml format. Does not return anything even if the format is the same as the api asks. Status 400 BAD REQUEST.

2:27 PM 13/02/2026 [HEAD /projects/ :id/ categories] Head request to categories of projects endpoint with specific project id and in xml format. Does not return anything even if the format is the same as the api asks. Status 400 BAD REQUEST.

requests in XML do not work!

2:28 PM 13/02/2026 [DELETE /projects/ :id/ categories] Delete request to category of projects endpoint with specific project id. Successfully deletes the correct relationship. Status 200 OK.

2:29 PM 13/02/2026 [DELETE /projects/ :id/ categories] Delete request to categories of projects endpoint with invalid project id. Fail to find the id as expected. Status 404 NOT FOUND.

2:29 PM 13/02/2026 [POST /categories/ :id/ projects] Post request to categories of projects endpoint with valid category id. Successfully creates a new relationship between the category with specific id and the todo with id specified in the body. Status 201 CREATED.

2:30 PM 13/02/2026 [GET /categories/ :id/ projects] Get request to the categories of projects endpoint with specific category id. Successfully returns the categories of the specific todo as expected. Status 200 OK. Note: it returns the relationships that were created with POST todos -> categories and POST categories -> todos

2:31 PM 13/02/2026 [HEAD /categories/ :id/ projects] Head request to categories of projects endpoint with specific category id. Return the category of the specific todo as expected. Status 200 OK.

2:32 PM 13/02/2026 [GET/categories/ :id/ projects] Get request to categories of projects endpoint without category id.This returns the list of todos of categories with status 200 OK. This is an undocumented API.

2:34 PM 13/02/2026 [GET /categories/ :id/ projects] Get request to categories of projects endpoint with invalid category id.This returns the list of todos of categories while it is not supposed to, with status 200 OK. This is a bug.

???bug

2:35 PM 13/02/2026 [HEAD /categories/ :id/ projects] Headrequest to categories of projects endpoint with specific category id. This returns a header while it is not supposed to, with status 200 OK. This is a bug.

???bug

2:35 PM 13/02/2026 [DELETE /categories/ :id/ projects] Delete request to categories of projects endpoint with specific category id. Successfully deletes the correct relationship. Status 200 OK.

2:36 PM 13/02/2026 [DELETE /categories/ :id/ projects] Delete request to categories of projects endpoint with invalid todo id. Fail to find the id as expected. Status 404 NOT FOUND.

Areas covered:

---

endpoints of the interoperability of todo and capabilities

Endpoints of the interoperability of projects and capabilities

** Ending session

Decisions Made During Session

Tested relationship endpoints between TODOs and CATEGORIEs as well as CATEGORIEs and PROJECTs, thus went a bit over 45 minutes.

Tested similar functionalities for both types of relationships, maily sticking to documented APIs.

Invalid ID bug is reoccurring, same on as the previous testing session, when an invalid ID is given with a get of header request and it returns data instead of an error.

After discovering JSON works but XML fails, decided to test XML format for both relationship endpoints.

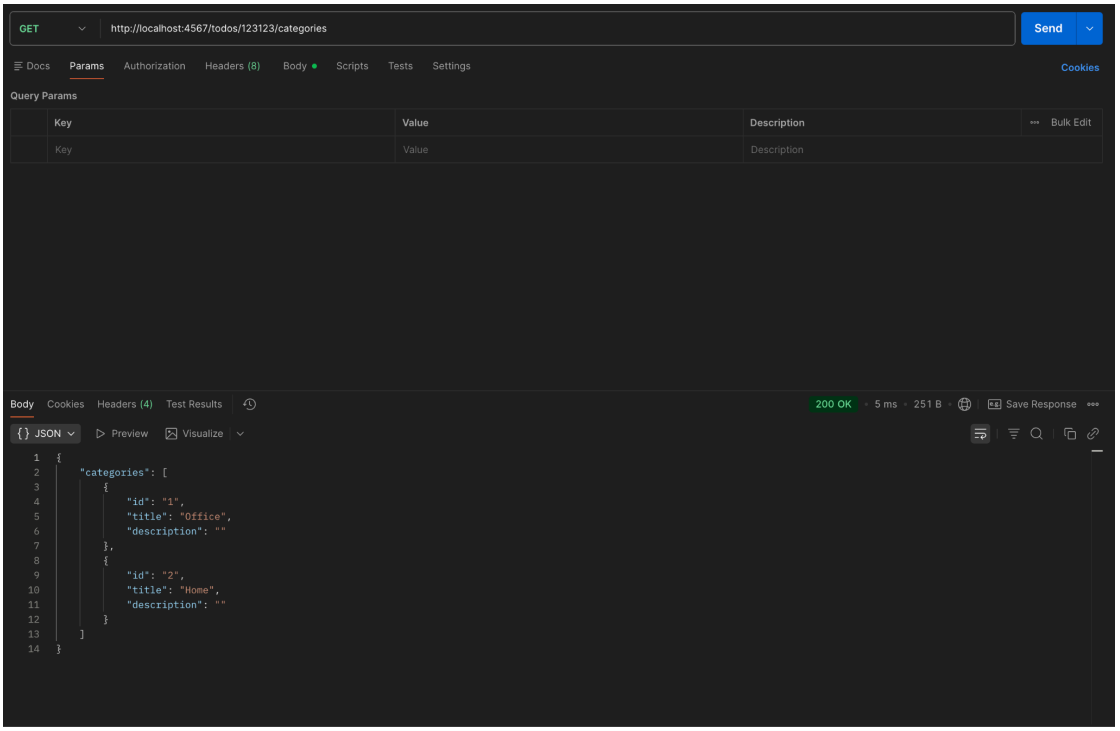
We tested with the JSON format first then the XML format but we realized that no XML request worked.

Bugs logged

---

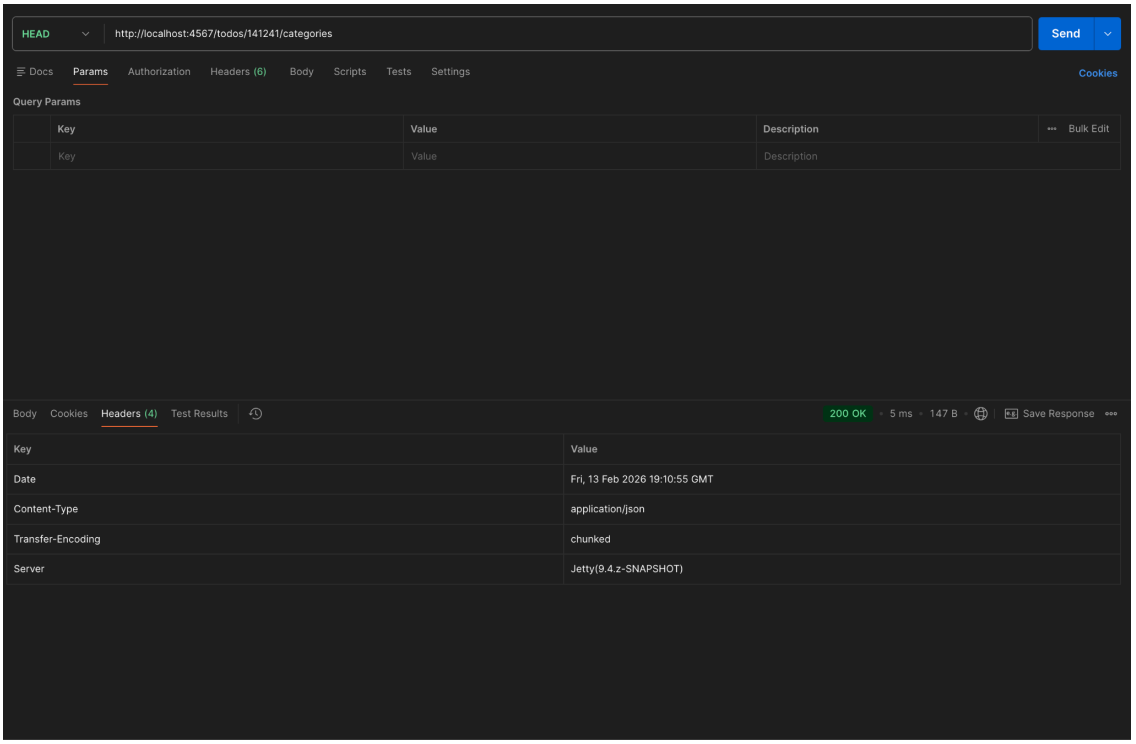
1) endpoint: GET /todos/:id/categories

Performing GET /todos/:id/categories with non-existing TODO ID (123123) returned data for all other valid IDs instead in the JSON message. This is incorrect behavior. Should return an error with corresponding status message. ##BUG (Refer to .docx)



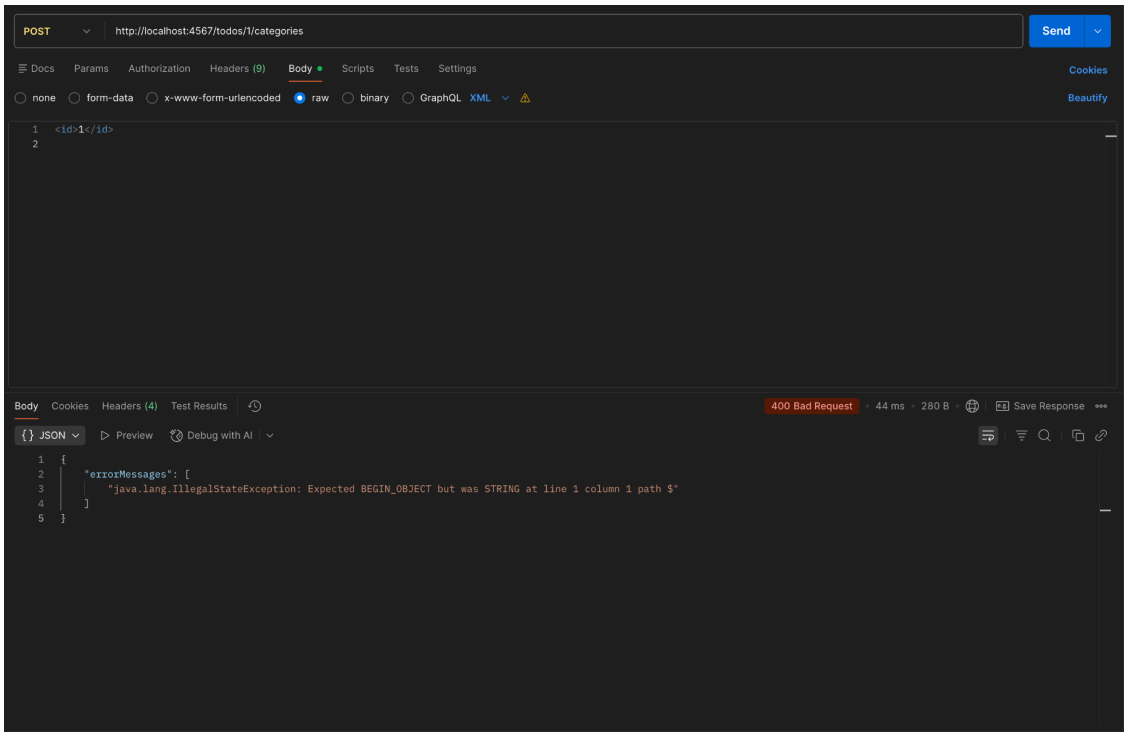
2) endpoint: HEAD /todos/:id/categories

Performing HEAD /todos/:id/categories with non-existing TODO ID (141241) returned header data in the JSON message. This is incorrect behavior. Should return an error with corresponding status message. ##BUG (Refer to .docx)



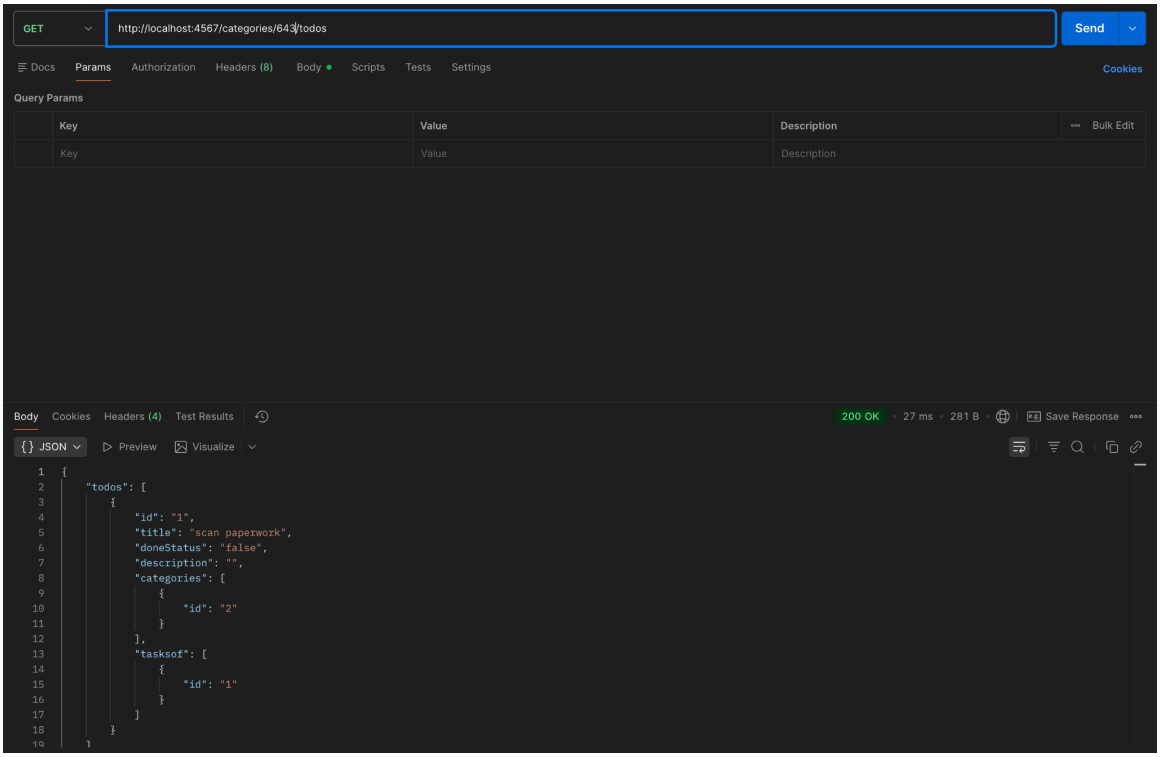
3) endpoint: POST /todos/:id/categories

Performing POST /todos/:id/categories with a valid TODO ID and correct format body in XML format (<id>1</id>) returned an error.



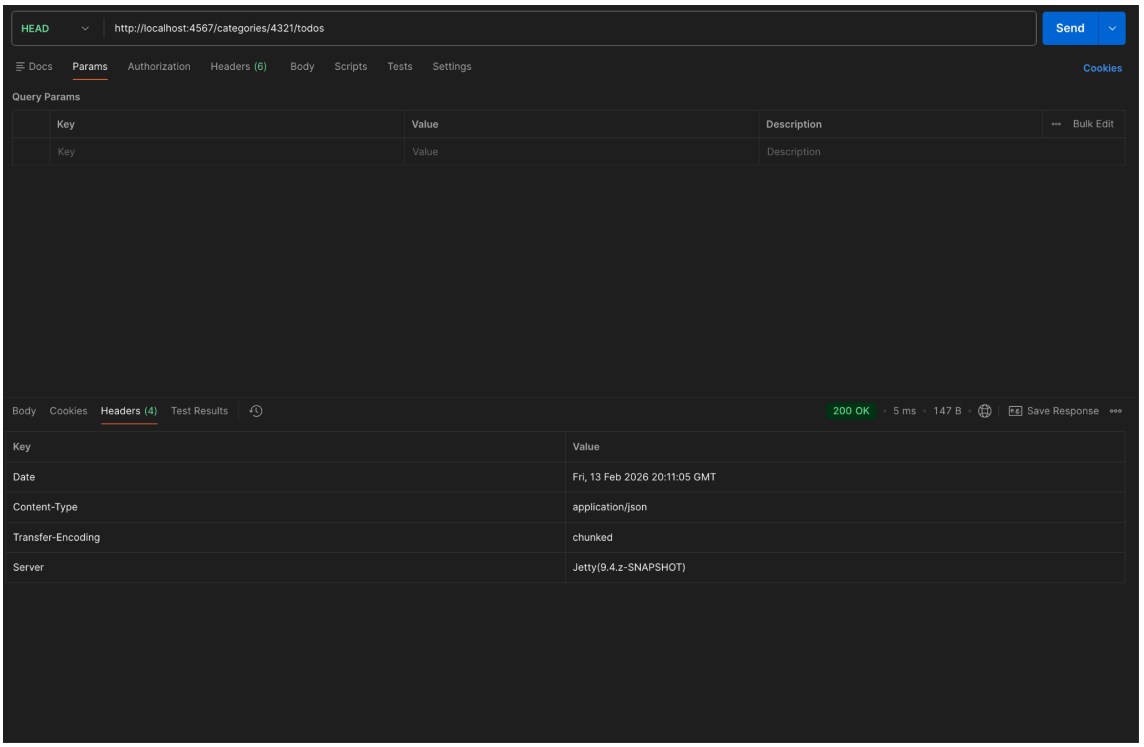
4) endpoint: GET /categories/:id/todos

Performing GET /categories/:id/todos with non-existing CATEGORIES ID (643) returned data for all other valid IDs instead in the JSON message. This is incorrect behavior. Should return an error with corresponding status message. ##BUG (Refer to .docx)



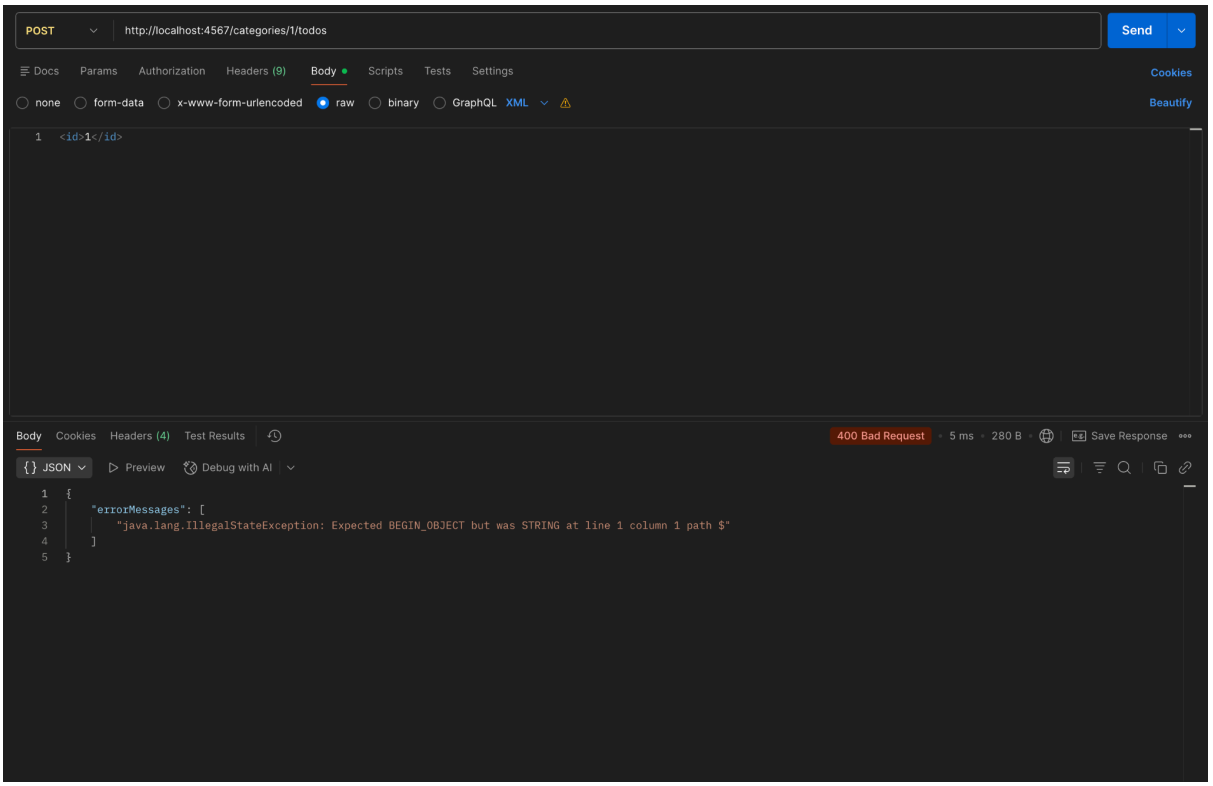
5) endpoint: HEAD /categories/:id/todos

Performing HEAD /categories/:id/todos with non-existing CATEGORY ID (4321) returned header data in the JSON message. This is incorrect behavior. Should return an error with corresponding status message. ##BUG (Refer to .docx)



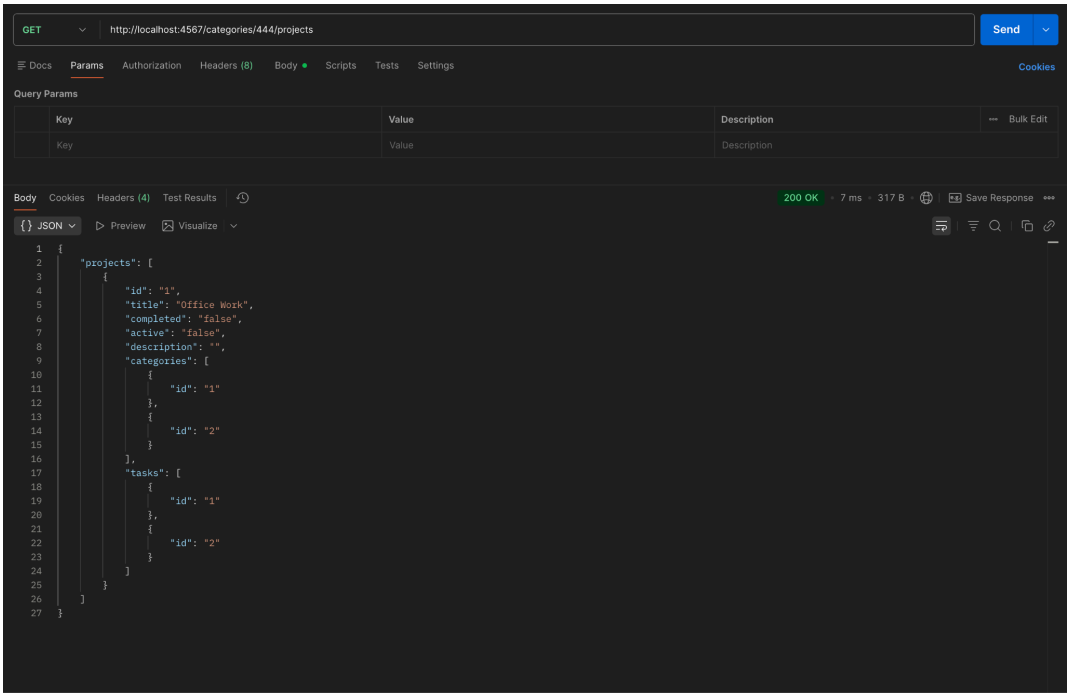
6) endpoint: POST /categories/:id/todos

Performing POST /categories/:id/todos with a valid CATEGORY ID and correct format body in XML format (<id>1</id>) returned an error.



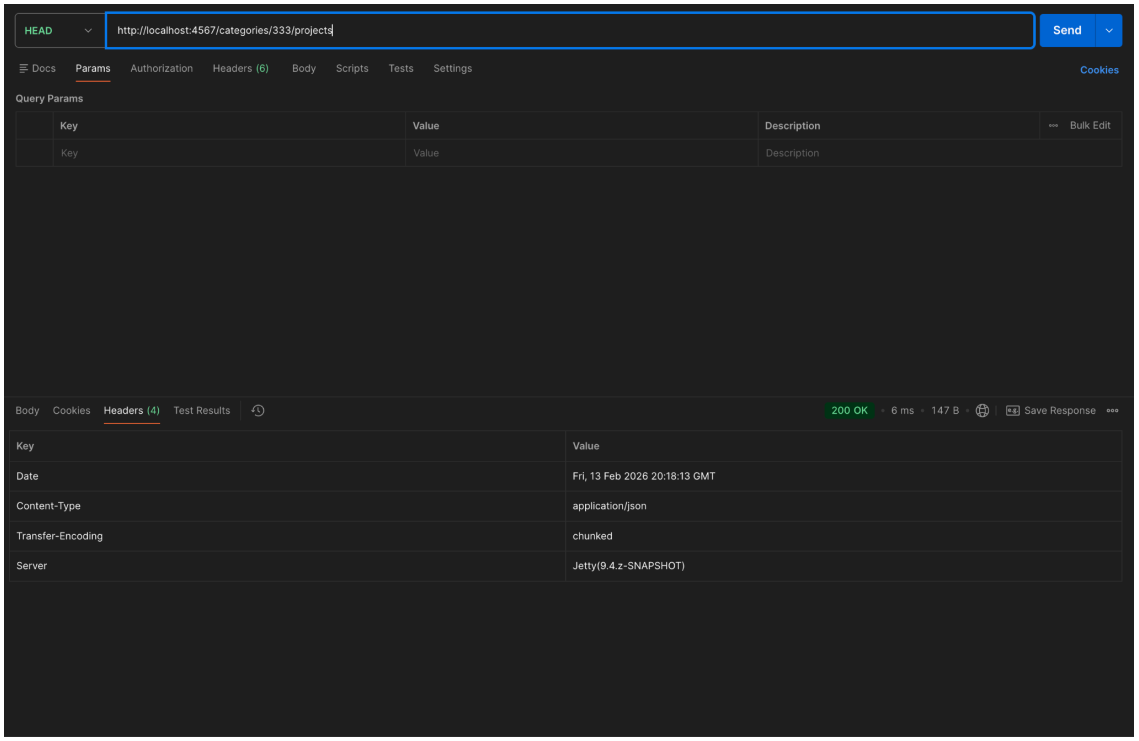
7) endpoint: GET /categories/:id/projects

Performing GET /categories/:id/projects with non-existing CATEGORIES ID (444) returned data for all other valid IDs instead in the JSON message. This is incorrect behavior. Should return an error with corresponding status message. ##BUG (Refer to .docx)



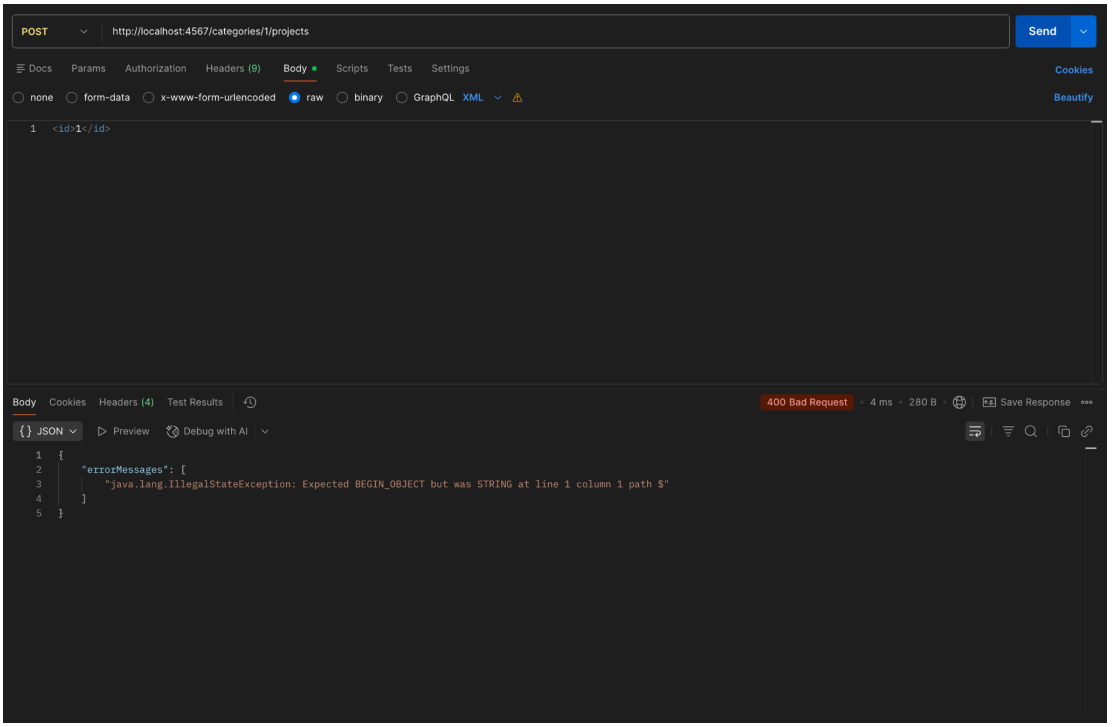
8) endpoint: HEAD /categories/:id/projects

Performing HEAD /categories/:id/projects with non-existing CATEGORY ID (333) returned header data in the JSON message. This is incorrect behavior. Should return an error with corresponding status message. ##BUG (Refer to .docx)



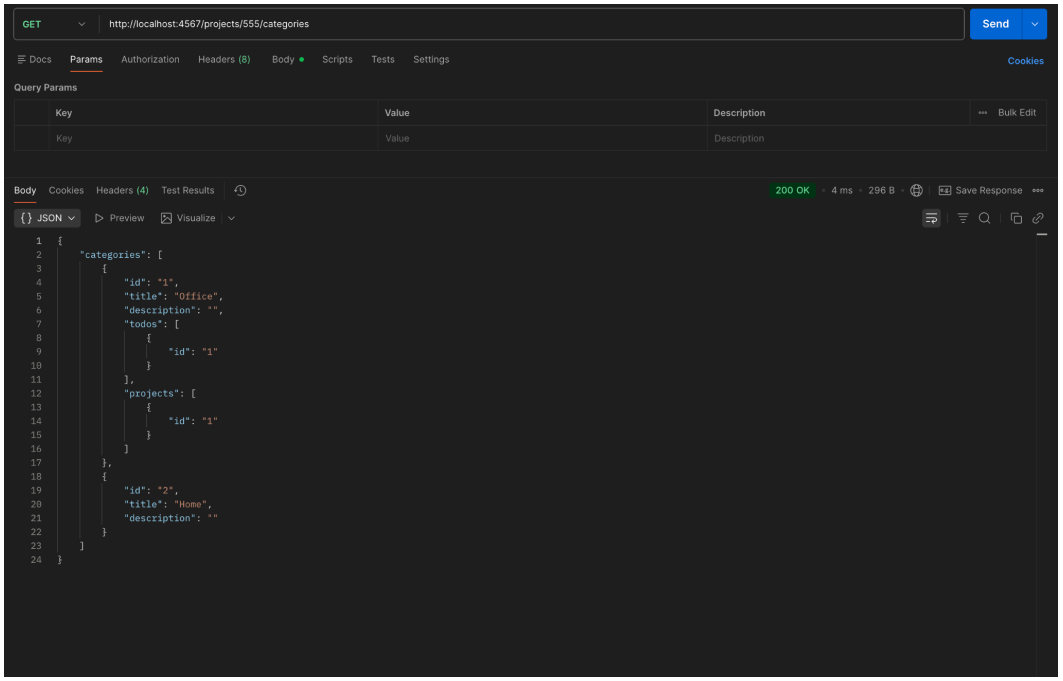
9) endpoint: POST /categories/:id/projects

Performing POST /categories/:id/projects with a valid CATEGORY ID and correct format body in XML format (<id>1</id>) returned an error.



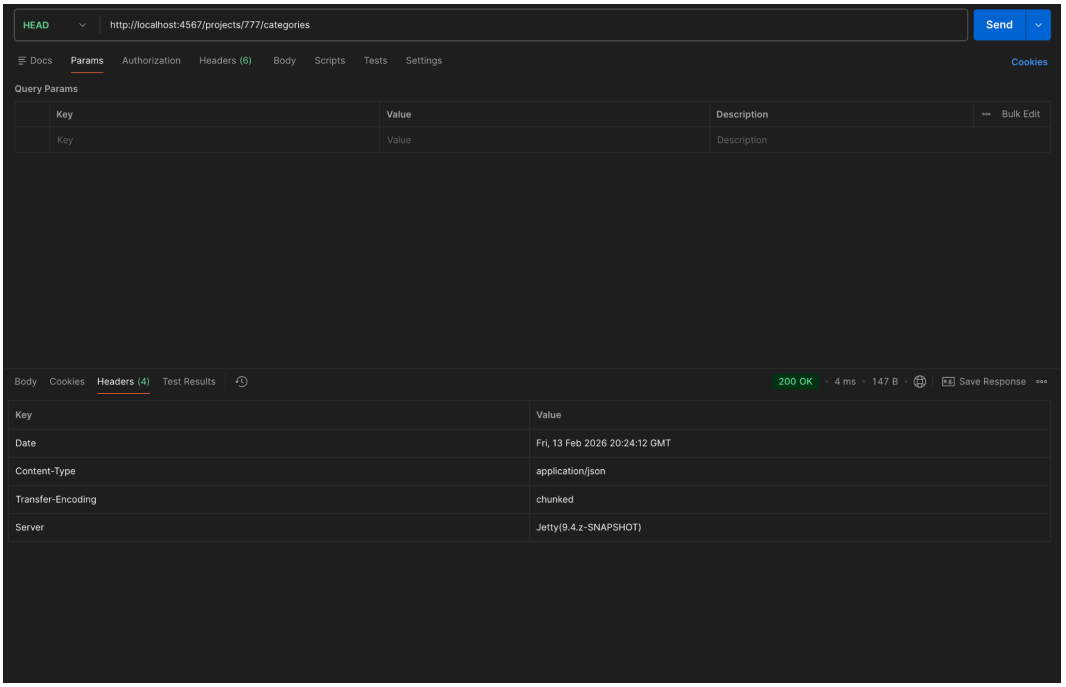
10) endpoint: GET /projects/:id/categories

Performing GET /projects/:id/categories with non-existing PROJECT ID (555) returned data for all other valid IDs instead in the JSON message. This is incorrect behavior. Should return an error with corresponding status message. ##BUG (Refer to .docx)



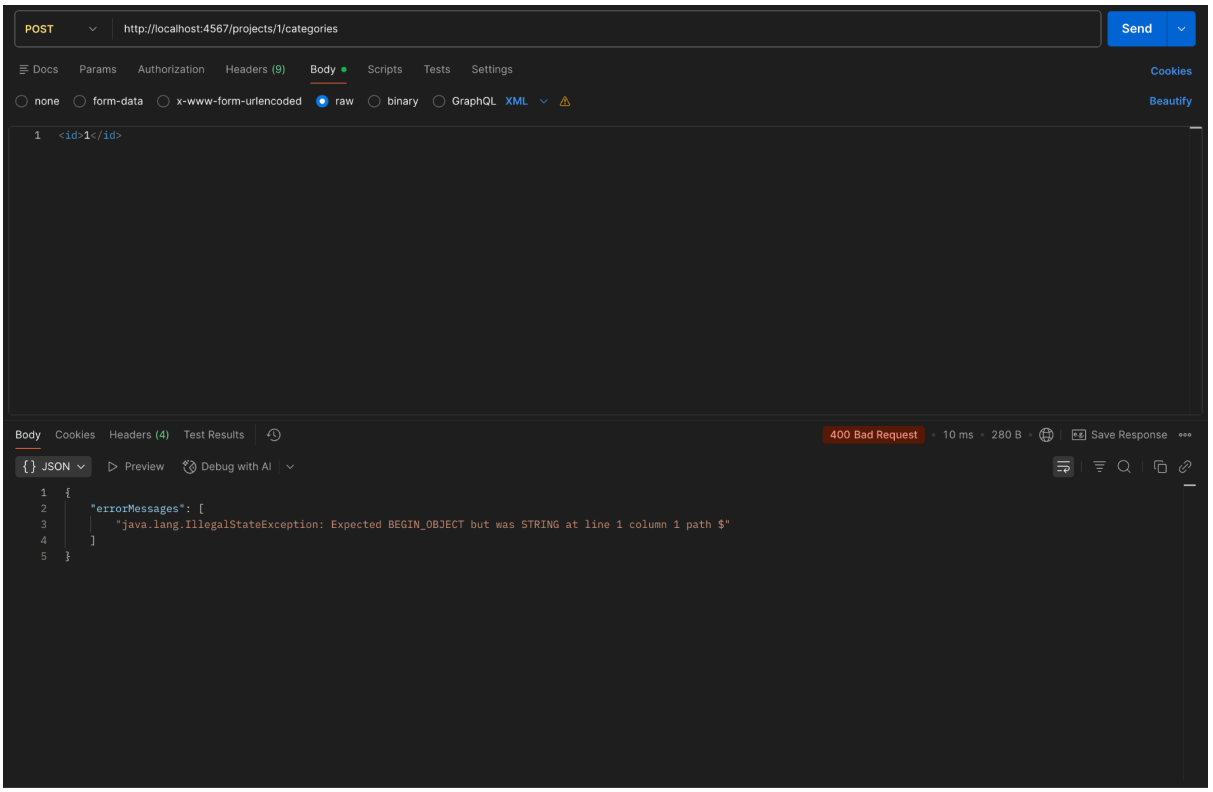
11) endpoint: HEAD /projects/:id/categories

Performing HEAD /projects/:id/categories with non-existing PROJECT ID (777) returned header data in the JSON message. This is incorrect behavior. Should return an error with corresponding status message. ##BUG (Refer to .docx)



12) endpoint: POST /projects/:id/categories

Performing POST /categories/:id/projects with a valid PROJECT ID and correct format body in XML format (<id>1</id>) returned an error.



Ideas/Plans for Future Sessions

Cover CRUD and HEAD operations for Todos and Categories

Test duplicate relationship creation (add same project twice)

Test deleting relationship twice

Test performance with 100+ todos

Test content-type mismatch (send XML header but JSON body)

Areas of Potential Risk

Similarly to the last sessions,

BUG #1,2,3,4,7,8,9 and 10 are data concerns, they return data for an invalid request. The user should not have access to this data.

BUG #5, 6, 11 and 12 are format handling risks. XML not parsed even though API suggests it is supported. Could indicate incomplete implementation.

Capabilities

CRUD operations on Projects/Capabilities, Capabilities/projects, todos/capabilities and capabilities/todos endpoints in JSON

HEAD operation on the same endpoints in JSON

Refer to Capabilities Script: projectsCapabilities.sh, todosCapabilities.sh and categoriesCapabilities.sh

Summary of session findings:

This session expanded relationship testing to cover projects/categories, todos/categories, and categories/todos endpoints. Twelve bugs were identified, continuing the patterns from Session 2: HEAD requests with invalid IDs return data instead of errors (bugs #1–4, #7–10),

GET requests with invalid IDs return incorrect data (bugs noted inline), and XML body parsing fails across all tested relationship POST endpoints (bugs #5–6, #11–12).

These recurring bugs across relationship endpoints represent a systemic risk: invalid requests leak data and XML support appears broadly unimplemented. The same data exposure and format-handling issues from Session 2 persist, suggesting they are not isolated defects.