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 Project

Exploratory testing for API endpoints

Environment

MacOS 26.3

Postman software

START

2:06PM 12/02/2026

TESTER

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

#DURATION

---

45 minutes

TEST NOTES

2:06 PM 12/02/2026 [GET /projects] Get request to projects endpoint. Returns a list of the current projects as expected.

2:11 PM 12/02/2026 [HEAD /projects] Head request for projects. Returns status 200 as expected.

2:13 PM 12/02/2026 [POST /projects] Post request to projects endpoint to add a new project to the list. We added an empty project and obtained a 201 CREATED status. ID, completed and active fields were given default values and title/desc fields are blank.

2:15 PM 12/02/2026 [GET /projects/ :id] Get request for project with specified id. Returns specific project as expected. Status 200 OK.

2:16 PM 12/02/2026 [GET /projects/ :id] Get request for project with specified id for a nonexistent project. Returns 404 NOT FOUND status.

2:20 PM 12/02/2026 [HEAD /projects/ :id] Head request for projects with specified ID. Returns status 200 as expected with header info (without body).

2:23 PM 12/02/2026 [POST /project/ :id] Post request to projects endpoint failed. Tried to create a new project with an unused id. Error status 400 BAD REQUEST: Invalid creation: Not allowed to create with id.

2:28 PM 12/02/2026 [POST /projects/] Post request to projects endpoint with a specified title and desc works as expected. Status 201 CREATED.

2:32 PM 12/02/2026 [POST /project/ :id] Post request to projects endpoint to update a project with specified ID. We were able to update the title, description and the active status as expected. Status 200 OK.

2:35 PM 12/02/2026 [PUT /projects/ :id] Put request to projects with id to update a project with specified ID. We were able to update the title, description and the active status as expected. Status 200 OK.

2:36 PM 12/02/2026 [PUT /projects/ :id] Put request to projects with invalid id. Fails as expected with status 404 NOT FOUND.

2:37 PM 12/02/2026 [PUT /projects/] Put request to projects with no id. Fails as expected with status 405 METHOD NOT ALLOWED.

2:38 PM 12/02/2026 [DELETE /projects/ :id] Delete request to projects endpoint with a specified ID. Deletes the project as expected with status 200 OK.

2:39 PM 12/02/2026 [DELETE /projects/ :id] Delete request to projects endpoint with a specified ID for a non-existing project. Returns an appropriate error message as expected: status 404 NOT FOUND.

2:39 PM 12/02/2026 [DELETE /projects/ :id] Delete request to projects endpoint with a specified ID for a non-existing project. Returns an appropriate error message as expected: status 404 NOT FOUND.

2:40 PM 12/02/2026 [DELETE /projects/] Delete request to projects endpoint with no id. Fails as expected with status 405 METHOD NOT ALLOWED.

2:43 PM 12/02/2026 [GET /projects/ :id / categories] Get request to project categories endpoint with id. Return empty as expected since no categories were created. Status 200 OK.

2:45 PM 12/02/2026 [POST /projects/ :id / categories] POST request to project categories endpoint with id.

Areas covered:

---

endpoints related to project

** Ending session

Decisions Made During Session

Focused on core Project CRUD + HEAD endpoints before testing relationships.

Established baseline with GET /projects before performing mutations.

Tested empty POST first to observe default value behavior.

Compared POST vs PUT to check update semantics.

Tested invalid IDs and missing IDs to verify validation and status codes.

Explored /projects/:id/categories to observe behavior with empty relationships.

Deferred relationship, concurrency, XML, and edge-case testing due to 45-minute time constraint.

Bugs logged

No bugs logged during this session

Ideas/Plans for Future Sessions

Test relationships related to projects

Test concurrency (simultaneous POST requests)

Test special characters in project names

Test malformed JSON body (missing brackets, wrong field name)

Discover potential undocumented API requests

Areas of Potential Risk

No areas of potential risk since no bugs were logged during this session

Capabilities

CRUD operations on Projects Entity in JSON

HEAD operation on Projects Entity in JSON

Refer to Capabilities Script: projectsCapabilities.sh

Summary of session findings:

This session focused on exploring the core CRUD and HEAD operations for the Projects entity in JSON format. All standard endpoints behaved as expected, with correct status codes for valid and invalid requests. No bugs were logged.

The session established a solid baseline for project endpoint behavior and identified key areas for follow-up, including relationship testing (e.g., /projects/:id/categories), concurrency, XML support, and edge cases like malformed JSON. These were deferred due to the 45-minute time constraint.