

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

```
java -jar runTodoManagerRestAPI-1.5.5.jar
```

Area

Capabilities and areas of potential instability (documented and undocumented) as well as tests
using data typical to the intended use of the application.

Environment

Windows 10, 1920x1080 (Joey)
MacOS 13.4.1, 3024 × 1964 (Ke)

Start

8:00pm, Tuesday October 11th 2023

Testers

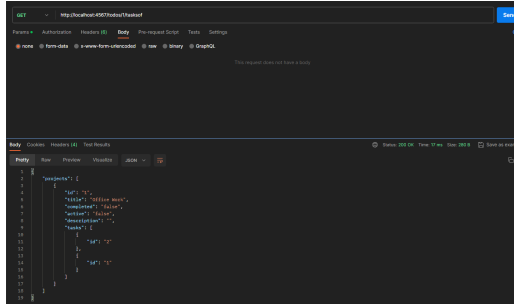
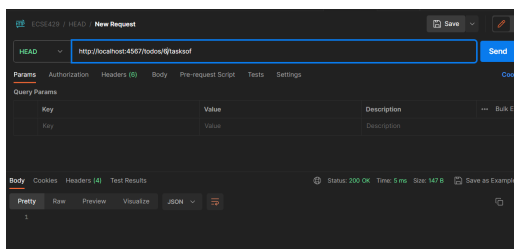
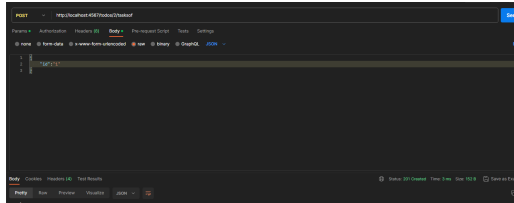
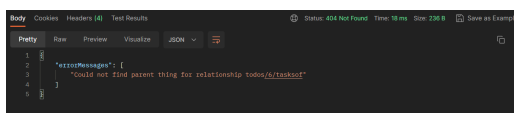
Ke Yan	260988148	ke.yan@mail.mcgill.ca
Joey Liu	260969992	lucas.liuliongwah@mail.mcgill.ca

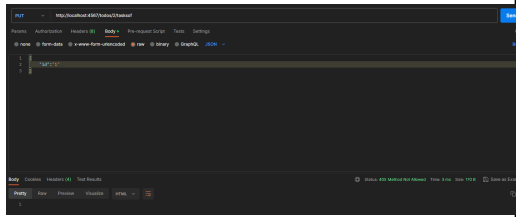
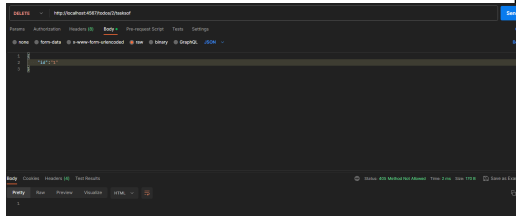
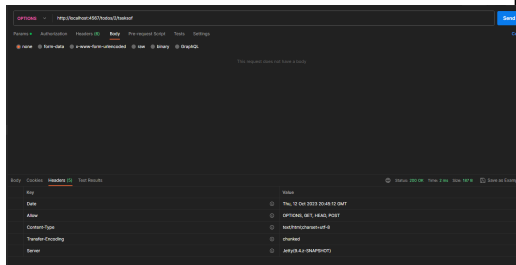
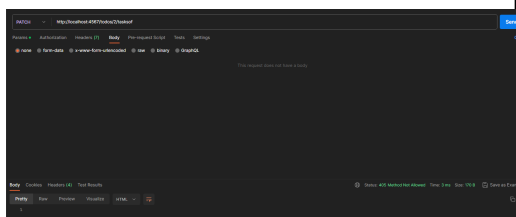
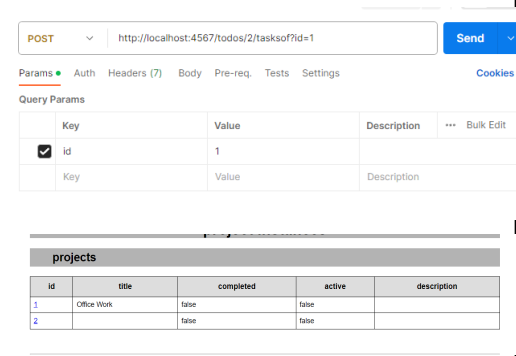
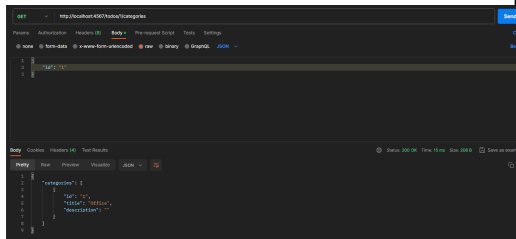
Session Breakdown

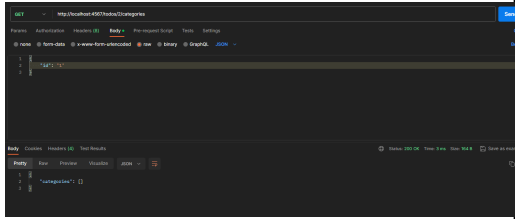
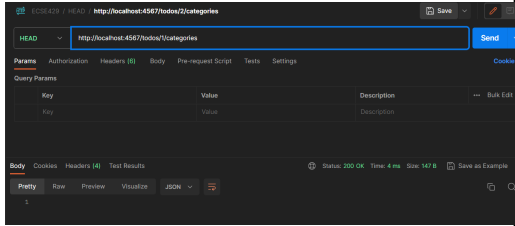
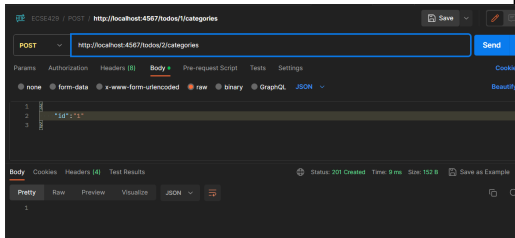
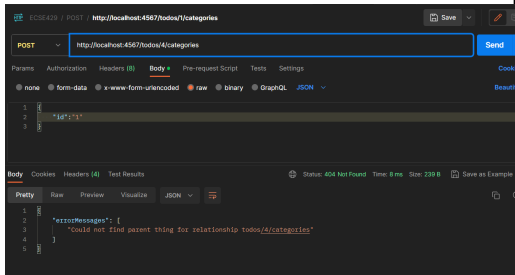
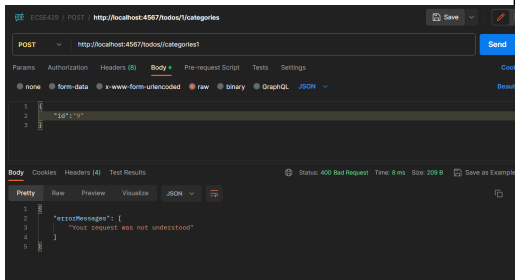
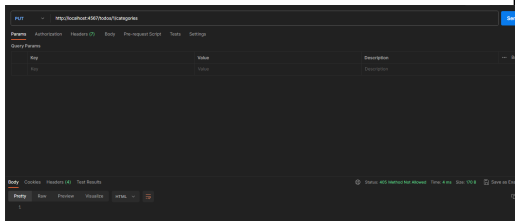
Total length 45 minutes

Test Notes

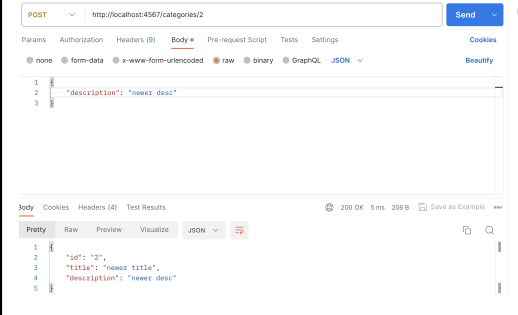
Session 2

Time	Tester	Request	Type	Comment	Screenshot
8:02	Joey	http://localhost:4567/todos/:id/tasksof	GET	Returns project related to Id 1. (Works as expected)	
8:03			GET (Id not in memory)	Returns project related to Id 3 which means none. However, no warnin that todo with id 3 doesn't exist, still returns 200 (Unexpected)	
8:05			HEAD	Received the response OK and code 200. (Works as expected)	
8:07			POST	Received the response OK and code 201, but instead description should have been "created the relationship" (Does not works as expected)	
8:07			POST (Id not in memory, todos Id: 6)	Received the response Not Found and code 404. (Works as expected)	

8:10			PUT (undocumented)	Received code 405 and “Method Not Allowed” as message. (Works as expected)																					
8:12			DELETE (undocumented)	Received code 405 and “Method Not Allowed” as message. (Works as expected)																					
8:13			OPTIONS (undocumented)	Received code 200 and “OK” as message. (Works as expected) We can see in the Allow header, all the endpoints that can be performed																					
8:15			PATCH (undocumented)	Received code 405 and “Method Not Allowed” as message. (Works as expected)																					
3:41			POST	Line query with id = 1, created a project in database with id 2. (Does not works as expected)	 <table><thead><tr><th colspan="5">projects</th></tr><tr><th>id</th><th>title</th><th>completed</th><th>active</th><th>description</th></tr></thead><tbody><tr><td>1</td><td>Office Work</td><td>false</td><td>false</td><td></td></tr><tr><td>2</td><td></td><td>false</td><td>false</td><td></td></tr></tbody></table>	projects					id	title	completed	active	description	1	Office Work	false	false		2		false	false	
projects																									
id	title	completed	active	description																					
1	Office Work	false	false																						
2		false	false																						
8:17		http://localhost:4567/todos/:id/categories	GET (Id : 1)	Returns category related to Id 1. (Works as expected)																					

8:18			GET (Id : 2)	Returns category related to Id 2. (Works as expected)	
8:20			HEAD	Received the response OK and code 200. (Works as expected)	
8:21			POST	Received the response OK and code 201, but instead description should have been “created the relationship” (Does not works as expected)	
8:23			POST (Id not in memory, todos Id: 4)	Received the response Not Found and code 404. (Works as expected)	
8:23			POST (Id not in memory, category Id: 9)	Received the response Not Found and code 404. (Works as expected)	
8:25			PUT (undocumented)	Received code 405 and “Method Not Allowed” as message. (Works as expected)	

8:27			DELETE (undocumented)	Received code 405 and “Method Not Allowed” as message. (Works as expected)	
8:28			OPTIONS (undocumented)	Received code 200 and “OK” as message. (Works as expected) We can see in the Allow header, all the endpoints that can be performed	
8:29			PATCH (undocumented)	Received code 405 and “Method Not Allowed” as message. (Works as expected)	
8:30		http://localhost:4567/todos	POST (malformed json body)	Received code 400 and “Bad Request” as message.	
8:32			POST (malformed json body)	Received code 405 and “Bad Request” as message.	
8:33	Ke	http://localhost:4567/categories/:id	GET	Received the existing objects with a 200 code (Works as expected)	

8:35			PUT (title)	Received the response OK and code 200 and received updated object. (Works as expected)	
8:36			PUT (description only)	Object cannot be updated, 400 error with message "title : field is mandatory" (Does not works as expected)	
8:38			PATCH	Received code 405 and “Method Not Allowed” as a message. (Works as expected)	
8:39			POST	Received the response OK and code 200 and received an updated object. (Works as expected)	
8:41			PUT	Received the response OK and code 200 and received an updated object. (Works as expected)	

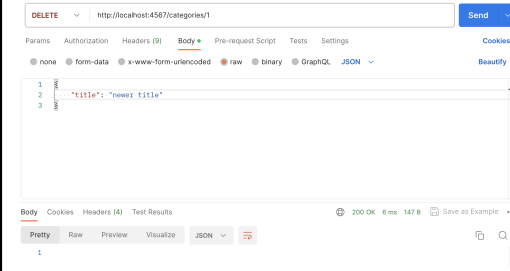
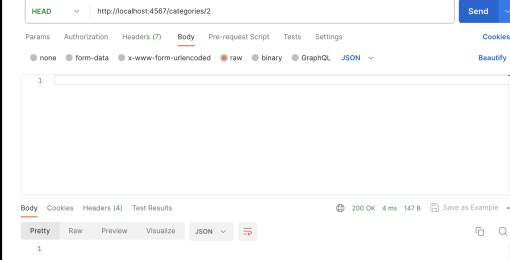
8:42			DELETE	Received the response OK and object is deleted (Works as expected)	
8:43			HEAD	Received the response OK and code 200, no output in body. (Works as expected)	

Table 2: Findings of Session 2

Summary of Session

This session we tested ...

todos/:id/tasksof

- GET
 - Id in memory **Works as expected**
 - Id not in memory **Unexpected**
- Head
- POST
 - Valid id **Does not works as expected**
 - Id in line query **Does not works as expected**
 - Invalid id **Works as expected**
- PUT (undocumented) **Works as expected**
- DELETE (undocumented) **Works as expected**
- OPTION (undocumented) **Works as expected**
- PATCH (undocumented) **Works as expected**

/todos/:id/categories

- GET **Works as expected**
- HEAD **Works as expected**
- POST **Does not works as expected**
 - Valid id
 - Invalid todo id **Works as expected**
 - Invalid categories id **Works as expected**
- PUT (undocumented) **Works as expected**
- DELETE (undocumented) **Works as expected**
- OPTION (undocumented) **Works as expected**

- PATCH (undocumented) **Works as expected**
- /todos
- POST
 - malformed json **Works as expected**
- /categories/:id
- GET **Works as expected**
 - PUT
 - Title only **Works as expected**
 - Title and description **Works as expected**
 - Description only **Does not work as expected**
 - PATCH **Works as expected**
 - POST
 - Title **Works as expected**
 - Description **Works as expected**
 - DELETE **Works as expected**
 - HEAD **Works as expected**

List of Concerns Identified in Session

- For todos/:id/tasks GET, if we try to get the tasks of a todo that does not exist (invalid id), an empty list is returned with code 200. However, the todo doesn't exist. This could mislead users into thinking the todo id is valid.
- POST for todos/:id/tasks, if we pass in an id in the query it may not be used in the created object
- When doing a POST for todos/:id/tasks and todos/:id/categories, a 201 code is returned, however the "created the relationship" description is missing
- For /categories/:id, PUT doesn't allow users to only edit the description of a category. They have to pass in a title as well. This isn't great if the user only wants to update the description
 - Additionally, the documentation for the PUT and POST endpoints are the same, but POST allows updates with only the description, which is confusing

List of new Testing Ideas Identified in Session

We noticed many issues caused by the use of invalid ids. However, due to time constraints, we were unable to test every endpoint with invalid id. It would be good to cover these during unit testing. Additionally, endpoints with two different sections, like todos and tasks, have a lot more noticeable instabilities. Some of our members will focus a whole session on interoperability to tackle those.