**Summary of Section Findings Todos Andrew Chirita**

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 for projects/id/tasks still obtains a valid response, which is the tasks for the first project with a valid id. The GET projects/ and GET projects/id are both documented capabilities that return data about all the projects or a specific project. As for the GET projects/id/tasks, it constitutes a potential area of instability because it returns a false positive JSON 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 valid response, when it shouldn’t. The HEAD projects/ and HEAD projects/id are both documented capabilities that return the header data about all the projects or a specific project. As for the HEAD projects/id/tasks, it constitutes a potential area of instability because it returns a false positive JSON 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. This means that each POST operation that creates a new project (POST /projects) constitutes an area of potential instability due to the project id being incorrectly increased. Every other POST operation is a documented capability.

The PUT operations behave similarly to the POST operations, however, if the json body omits a project field, it will reset that field to default. Therefore, this operation can be considered as an area of potential instability. It is also an undocumented capability, because it is not mentioned in the API documentation that the PUT operation can overwrite values other than those specified in the JSON input.

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. In that sense, each DELETE operation for the project section of the API is a documented capability.

LIST OF CONCERNS

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

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

HEAD and GET operations for tasks based on nonexistent project ids still return valid responses

PUT operations reset a project field to default if they are not included in the json request body