Bug Summary Considerations

1.

a.     Executive summary of bug in 80 characters or less.

Calling PUT/todos/:id sets the body fields without a value to their default values.

b.     Description of bug

When calling PUT/todos/:id without description and/or doneStatus for a todo with a valid id. Instead of having to not amend those fields, the program sets description to “” and done Status to false.

c.     Potential impact of bug on operation of system.

If a user decides to amend with the PUT command, the user can lose data that he previously had.

               d. Steps to reproduce the bug

For the following API call<http://localhost:4567/todos/1> with a PUT where 1 is a valid id of a todo set as inputs for the JSON only the title or at least one of description/doneStatus missing.

2.

a.     Executive summary of bug in 80 characters or less.

Calling PUT/todos/:id without a title field in the JSON crashes the app.

b.     Description of bug

When calling PUT/todos/:id without title for a todo with a valid id and regardless for the other fields, the app crashes instead of amending the todo with the description and/or doneStatus. This is not the case for the corresponding POST operation.

c.     Potential impact of bug on operation of system.

If a user decides to amend with the PUT command a todo by not wanting to change the title, the operation will never be allowed.

               d. Steps to reproduce the bug

For the following API call<http://localhost:4567/todos/1> with a PUT where 1 is a valid id of a todo set as inputs for the JSON only at least one of description/doneStatus. Don’t include the title.

3.

a.     Executive summary of bug in 80 characters or less.

Calling GET /todos/:id/tasksof with an invalid todos id returns a valid response

b.     Description of bug

When calling GET /todos/:id/tasksof with an invalid project id, a valid JSON response is returned with the tasks of the todos with the lowest valid id.

c.     Potential impact of bug on operation of system.

If a user decides to get the todos tasks of a todos with an invalid id, they will receive a wrong JSON response and think that the tasks in the JSON response actually correspond to the todos with the inputted id.

               d. Steps to reproduce the bug

For the following API, call [http://localhost:4567/todos/10000/tasks](http://localhost:4567/todos/1)of with a GET where 10000 is an invalid id of a todos, and do not include any values for the fields in the JSON body.

4.

a.     Executive summary of bug in 80 characters or less.

Calling HEAD /todos/:id/tasksof with an invalid todos id returns a valid response

b.     Description of bug

When calling HEAD /todos/:id/tasksof with an invalid todo id, a valid JSON response is returned.

c.     Potential impact of bug on operation of system.

If a user decides to get the todos tasks header information of a todo with an invalid id, they will receive a wrong JSON response and think that the tasks in the JSON response actually correspond to the todos with the inputted id.

               d. Steps to reproduce the bug

For the following API, call [http://localhost:4567/todos10000/tasks](http://localhost:4567/todos/1)of with a HEAD where 10000 is an invalid id of a todos, and do not include any values for the fields in the JSON body.

5.

a.     Executive summary of bug in 80 characters or less.

Calling PUT/projects/:id with missing fields in the JSON request resets the field value to default.

b.     Description of bug

When calling PUT/projects/:id with missing fields for a project with a valid id, the operation amends missing fields with default values, which are empty strings for strings, and false for booleans. This is not the case for the corresponding POST operation.

c.     Potential impact of bug on operation of system.

If a user decides to amend with the PUT command a project to change the title, they would have to input the title in the JSON request body.

               d. Steps to reproduce the bug

For the following API, call [http://localhost:4567/projects/1](http://localhost:4567/todos/1) with a PUT where 1 is a valid id of a project, and do not include any values for the fields in the JSON body.

6.

a.     Executive summary of bug in 80 characters or less.

Calling GET /projects/:id/tasks with an invalid project id returns a valid response

b.     Description of bug

When calling GET /projects/:id/tasks with an invalid project id, a valid JSON response is returned with the tasks of the project with the lowest valid id.

c.     Potential impact of bug on operation of system.

If a user decides to get the project tasks of a project with an invalid id, they will receive a wrong JSON response and think that the tasks in the JSON response actually correspond to the project with the inputted id.

               d. Steps to reproduce the bug

For the following API, call [http://localhost:4567/projects/10000/tasks](http://localhost:4567/todos/1) with a GET where 10000 is an invalid id of a project, and do not include any values for the fields in the JSON body.

7.

a.     Executive summary of bug in 80 characters or less.

Calling HEAD /projects/:id/tasks with an invalid project id returns a valid response

b.     Description of bug

When calling HEAD /projects/:id/tasks with an invalid project id, a valid JSON response is returned.

c.     Potential impact of bug on operation of system.

If a user decides to get the project tasks header information of a project with an invalid id, they will receive a wrong JSON response and think that the tasks in the JSON response actually correspond to the project with the inputted id.

               d. Steps to reproduce the bug

For the following API, call [http://localhost:4567/projects/10000/tasks](http://localhost:4567/todos/1) with a HEAD where 10000 is an invalid id of a project, and do not include any values for the fields in the JSON body.