

Their implementation was done well. They used everything they should have, and had clean code. Unfortunately, their package.json is missing dependencies, so I wasn't able to start the node without installing the dependencies manually (not by simply running npm install).

GET (/get)

The screenshot shows the API Network tab in VS Code. The request is a GET to `http://localhost:3000/get`. The Params tab is active, showing query parameters:

KEY	VALUE
author	yeah
alt	fhewifwe
tags	feriuwfw
image	notalink
description	fewjnfwefw
Key	Value

The Body tab is also active, showing the JSON response in Pretty format:

```
7  {
8    "image": "https://upload.wikimedia.org/wikipedia/commons/9/9d/Sir_Iim_Berner",
9    "description": "The internet and the Web aren't the same thing."
10 }
11 {
12   "id": 2,
13   "author": "Grace Hopper",
14   "alt": "Image of Grace Hopper at the UNIVAC I console",
15   "tags": "programming,linking,navy",
16   "image": "https://upload.wikimedia.org/wikipedia/commons/3/37/Grace_Hopper_a",
17   "description": "Grace was very curious as a child; this was a lifelong trait
18                 how an alarm clock worked and dismantled seven alarm clocks before her m
19                 limited to one clock)."
```

Get works as expected. The response gave me all the data from the gallery in json format.

Documentation for get was done well, not much to explain there. I tried out the 3 status codes (200, 204, and 404), they all worked.

GETBYID (getById/:id)

The screenshot shows a REST client interface with the following components:

- Top Bar:** Home, Workspaces, API Network, Explore.
- Overview Tab:** Displays the request URL `http://localhost:3000/getById/1`.
- Request Method:** GET.
- Params Tab:** Shows Query Params with the following table:

KEY	VALUE
<input type="checkbox"/> author	yeah
<input type="checkbox"/> alt	fhewifwe
<input type="checkbox"/> tags	feriuwfw
<input type="checkbox"/> image	notalink
<input type="checkbox"/> description	fewjnfwefw
Key	Value

- Body Tab:** Shows the response body in JSON format (Pretty view):

```
1 {
2   "id": 1,
3   "author": "Tim Berners-Lee",
4   "alt": "Image of Berners-Lee",
5   "tags": "html,http,url,cern,mit",
6   "image": "https://upload.wikimedia.org/wikipedia/commons/9/9d/Sir_Tim_Berners-Le",
7   "description": "The internet and the Web aren't the same thing."
8 }
```

The interface also includes tabs for Cookies, Headers (8), and Test Results, and a status bar at the bottom with Online, Find and Replace, and Console options.

Get by id works as expected. I tried getting the author by id one, and it returned just that, again in json format.

Documentation for get by id is done well. Parameters and status codes are correctly documented.

PUT (/put)

Home

Workspaces

API Network

Explore

PUT http://localhost:3000/put

GET http://localhost:3000/

http://localhost:3000/put?author=yeah&alt=fhewifwe&tags=feriuwfw&image=notalink&description=fewjnfwe

PUT

http://localhost:3000/put?author=yeah&alt=fhewifwe&tags=feriuwfw&image=notalink&descrip

Params

Authorization

Headers (9)

Body

Pre-request Script

Tests

Settings

Query Params

	KEY	VALUE
<input checked="" type="checkbox"/>	author	yeah
<input checked="" type="checkbox"/>	alt	fhewifwe
<input checked="" type="checkbox"/>	tags	feriuwfw
<input checked="" type="checkbox"/>	image	notalink
<input checked="" type="checkbox"/>	description	fewjnfwe
<input checked="" type="checkbox"/>	id	2
	Key	Value

Body

Cookies

Headers (8)

Test Results

Pretty

Raw

Preview

Visualize

JSON

1

{ }

Online

Find and Replace

Console

Home Workspaces API Network Explore

PUT http://localhost:3000/ GET http://localhost:3000/

http://localhost:3000/get

GET http://localhost:3000/get

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON

```

7      "image": "https://upload.wikimedia.org/wikipedia/commons/9/9d/Sir_Ilm_Berner
8      "description": "The internet and the Web aren't the same thing."
9    },
10   {
11     "id": 2,
12     "author": "Grace Hopper",
13     "alt": "Image of Grace Hopper at the UNIVAC I console",
14     "tags": "programming,linking,navy",
15     "image": "https://upload.wikimedia.org/wikipedia/commons/3/37/Grace_Hopper_a
16     "description": "Grace was very curious as a child; this was a lifelong trait
                    how an alarm clock worked and dismantled seven alarm clocks before her m
                    limited to one clock)."
17   }
18 ]

```

Online Find and Replace Console

PUT did not work for me. I tried changing the author under the id 2, but after using GET after using PUT, it did not show an updated author. Not sure what's the issue there, might be that I did not install all the dependencies properly, but as I mentioned before, this is their fault, having not configured the package.json file properly.

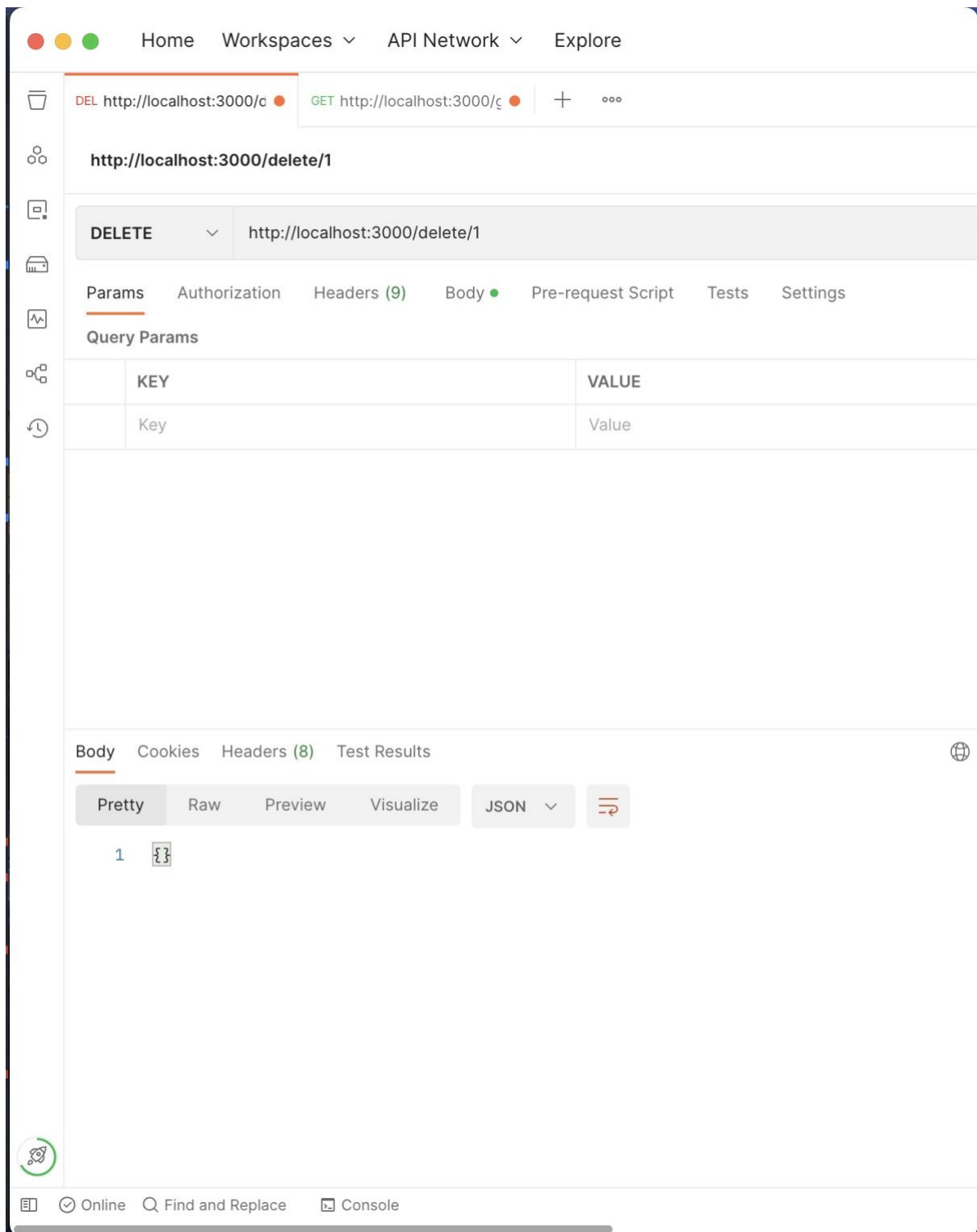
POST (/post)

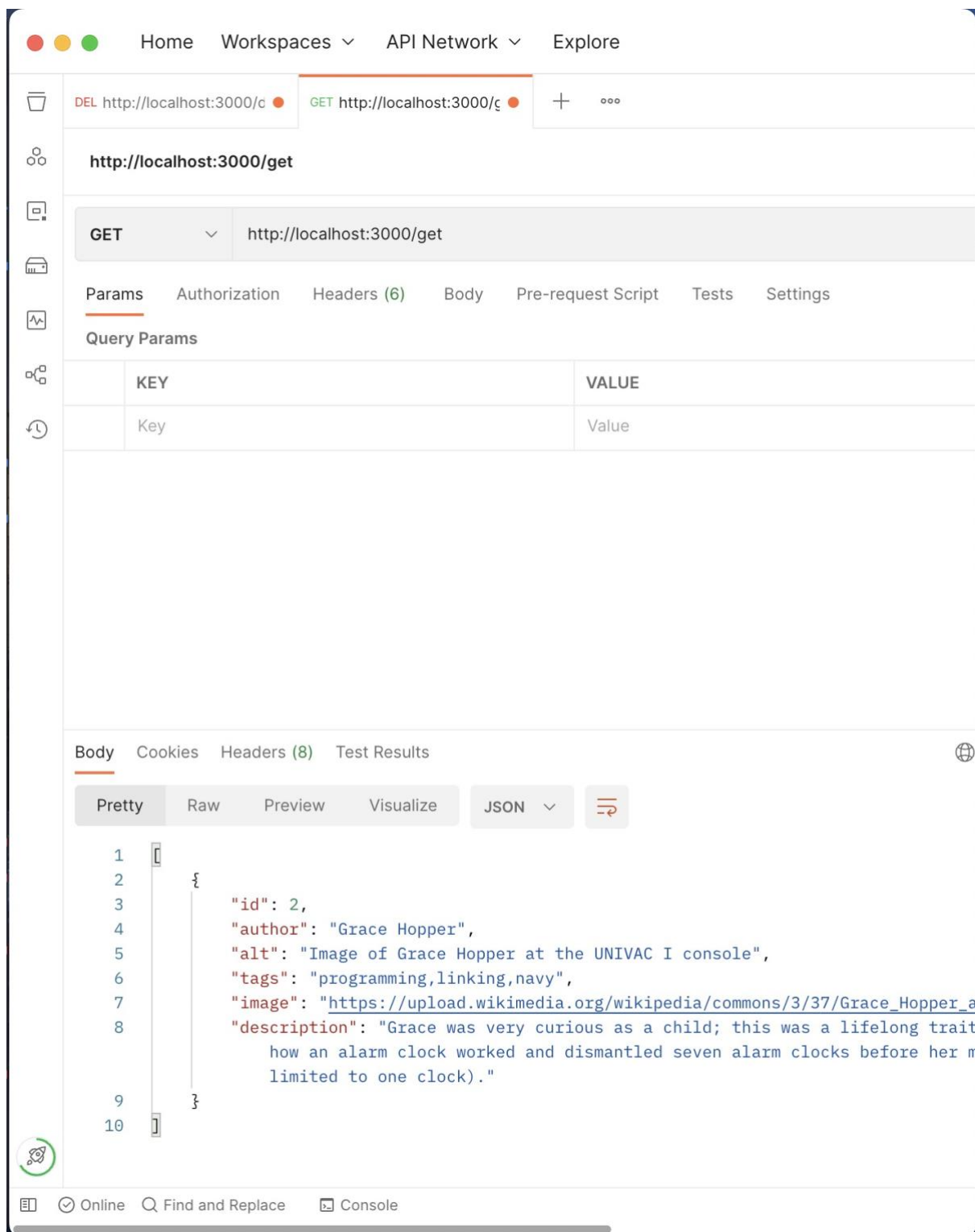
```
9      },
10     {
11       "id": 3,
12       "author": "kek",
13       "alt": "idk",
14       "tags": "yes",
15       "image": "thnotthislinkis",
16       "description": "some desc"
17     }
18   ]
}
```

Online Find and Replace Console

Post worked as expected. I added a new author, and it showed up when calling /get. Documentation is very accurate, and parameters and status codes work as expected.

DELETE (/delete/:id)





I managed to delete the author under the id "1" with no issues. Documentation is done perfectly.

Bonus:

Yes, the bonus was implemented into their second assignment. Everything seems to work properly, I tested adding a new author (not sure why it's called "open form"), update, and delete functions, and they all worked. Reset could've left the author that you guys added by default, instead they were all deleted. Not sure if that is intended, or correct behaviour.

Feedback:

Your documentation has single line comments on top of multi line comments, which makes it a bit unreadable in VS code, consider removing the one line `//`. I would also recommend adding proper dependencies in your `package.json` file, so users can just run `npm install` and it will work. I would also remove the `tempcoderunnerfile.js` from the server folder, as it now serves no purpose, and thus should not be in the folder. I would also consider leaving the default 2 authors in the table when resetting it, so your table does not break, and it just feels like more intended behaviour.

Conclusion:

Overall, your documentation, besides the double commenting, was done very well, you explained every call in detail mentioning every endpoint, parameter, and status code. Good job! I did not realise that we had to list the parameters's types and names in the documentation, so that surprised me a bit. You guys had it better than us, good job.