Documentation

Getting Started:

Please clone the repository:
https://github.com/Jack-Gitter/SentinelGroupCodingAssessment

Installing dependencies:

Navigate to the SentinelGroupCodingChallenge folder, and run the following command to install dependencies:

```
npm install
```

After the dependencies are done downloading, the server should be ready to run!

Running The Server

Please navigate to the SentinelGroupCodingChallenge folder, and run the following command

```
npx ts-node Server/Server.ts
```

After a short moment, you should see the following message pop up in the console:

“listening on port 8080”

This means that the server is live!

Running Unit Tests:

Because the server is not hosted, you will have to run the server with the instructions provided above to run the Endpoint.test.ts file. For all other files, you can run them without running the server.

After the server is running, open a new terminal and navigate to the Unit folder under Tests. Any test can then be run by running the following command

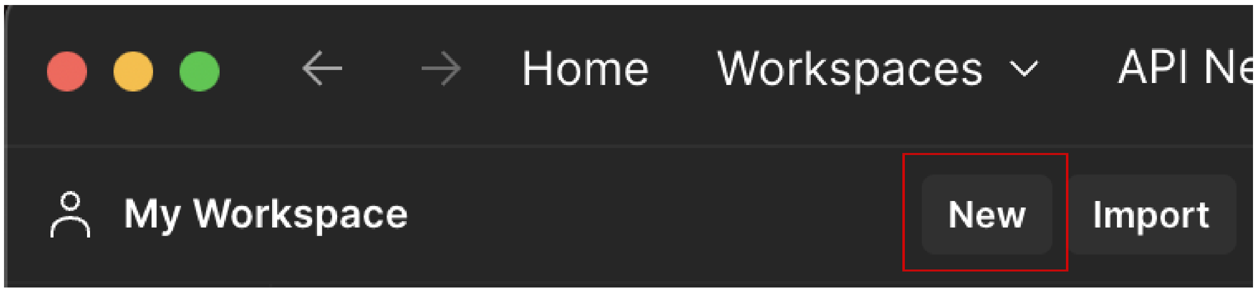
```
npm test – “file_name.test.ts”
```

If the "Endpoint.test.ts" file is run more than once in a row, it will error as the server has only allotted a certain number of vacation homes, and running the test suite twice in a row will request more resources than the server has to offer. Normally there would be cleanup after each test, although the API only required that we allow users to make reservations and not cancel them, so creating an entire new endpoint for cleanup seemed unnecessary.

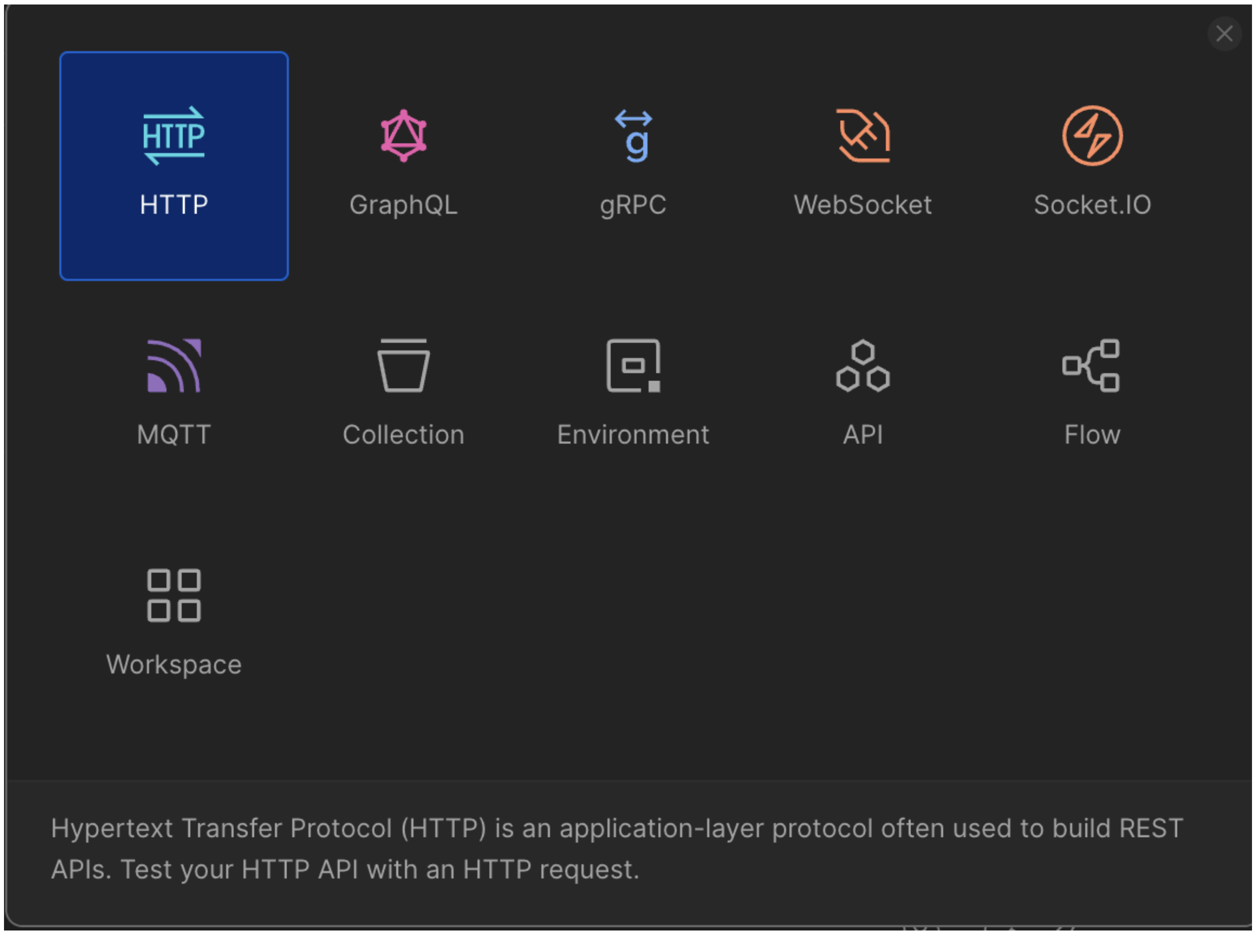
Running and using the API separately from the unit tests

Personally, I find the best way to test an HTTP endpoint is to use the desktop application called postman. It can be found via download here: https://www.postman.com/downloads/

After downloading and logging in, a new workspace can be created by clicking the "New" button in the top left

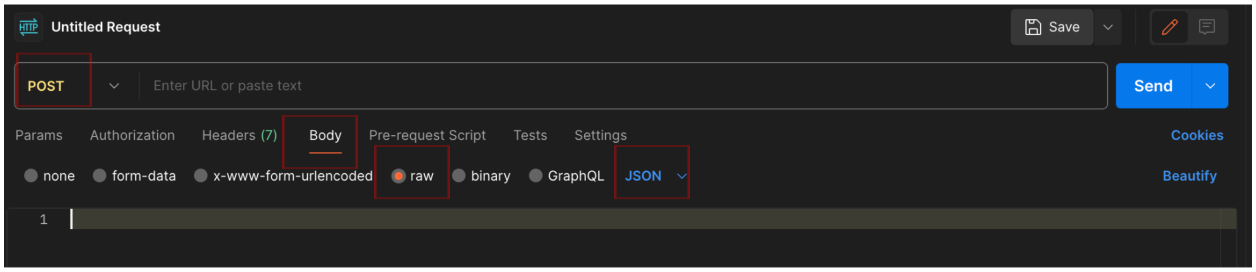


Please select HTTP from the options that appear



Hypertext Transfer Protocol (HTTP) is an application-layer protocol often used to build REST APIs. Test your HTTP API with an HTTP request.

After this, please switch the request type to POST, and navigate to the body tab and select “raw” and select JSON from the dropdown menu



If the server is running, you are now ready to make requests! Requests require the following body fields formatted in JSON:

```
{
        name: string
        propertyType: PropertyType (can be found in the Model/Types.ts file)
        startDate: string
        endDate: string
}
```

The startDate and endDate fields are parsed by the Date() class built into Typescript, and as such there is a standard for what strings are acceptable. That can be found here

https://tc39.es/ecma262/multipage/numbers-and-dates.html#sec-date-time-string-format

Ending Thoughts

This project was a lot of fun! I love working with APIs and really enjoyed making the model and programming this project. I used express for ease of use for server development, and axios to test my endpoints. I have tested the running instructions myself, but in case I forget anything because I'm operating on my own machine please reach out and I can help troubleshoot for sure! My email is jack.a.gitter@gmail.com.

I really look forward to continuing my interview process, thank you for the opportunity!