Oowlish JavaScript Challenge

Dear Candidate,

First of all, we are happy you got this far on our hiring process, congratulations! We created this small challenge to give you an opportunity to show your coding skills. We hope you enjoy it as much as we do. You have, roughly, up to one week to send it in.

The Task

You just received a task for fixing a HTML page `index.html`, but you **don't** have access to the main HTML file, or can change how the data is served from the server. The only resources you can use are CSS and JavaScript. The page consists of informations about the clinic doctors and their availability for appointments. There is only a list of doctors with their information.

Our operators team prepared a list of enhancements/bug fixes for the page:

1. The Available doctors filter is not working.
2. I can't set doctors as available and unvailable, the button is not working.
3. I find it difficult to see whether a doctor is available or unavailable. Can you add a "style" to make it easier to differentiate between available and unavailable doctors?
4. Would be great if we could have a field for searching doctors by name or its Unique Physician Identification Number (UPIN).

Also, we have a checklist from our software architect:

1. Use `searchContainer` div to add the search input component.
2. I noticed zip code and city information are not coming in our ajax API. Would be great if we could map this information so we don't lose it when filtering/searching. Keep that in mind when delivering the code.
3. It's important to have commits with clear description and separated by delivered context.
4. Can you add some automated tests for this page?

We know this list looks like a lot of work, but the team wants to see as many bullet points resolved as possible up to tomorrow EOD. Are you the person to finish this task?

You already have a server running into `http://localhost:3030/`. The server is based on `json-server` (https://github.com/typicode/json-server) and already has all the data you need to complete this task. You are not allowed to add new entities or modify the data structure of the already existing ones, but you are good to change/update any fields you think are suitable. You can also use any resources `json-server` provides for filtering/searching doctors in the api.

Requirements:

- One-command setup:
  - There should be no extra steps when running your app/server; either use an in-memory db, plain JSON server, or (if you have a full-blown SQL/NoSQL database) at the very least make it so one can run your DB instance using a single docker/docker-compose command.

    - The above also goes for environment variables/ `.env` files; if you do need to have a `.env` file, provide a sample `.env.example`.
- Unit test.

Our review

The focus of the review is on answering some questions such as:

- Does the application do what it promises?
- How many requirements were delivered?
- How clearly can you separate concerns in your code?
- Does your solution contain any dead/redundant code?
- How good was the UX solution applied for this test?
- Is your application bundle ready for production?
- Did you write good/meaningful tests?
- Did you add any documentation?
- Can we find bugs or minor flaws?
- Did you provide a complete product ready to deploy?
- Did you persist data changes to the database?

The following things most probably will not impress us:

- Overengineering;
- Typos, grammar mistakes in documentation and code.
- Changing index.ejs file or data.json

Extra points for:

- Well written documentation.
- Automated Testing.
- Using some modern javascript framework (React, Angular, VueJS)

Having that said, good luck and happy coding!