Problem 3 – Sprint Board

Working with Remote Data

For the solution of some of the following tasks, you will need to use an up-to-date version of the **local REST service** provided in the lesson's resources archive. You can read the documentation here.

Environment Specifics

Please be aware that every JS environment may **behave differently** when executing code. Certain things that work in the browser are not supported in **Node.js**, which is the environment used by **Judge**.

The following actions are **NOT** supported:

- `.forEach()` with **NodeList** (returned by `querySelector()` and `querySelectorAll()`)
- `.forEach()` with **HTMLCollection** (returned by `getElementsByClassName()` and `element.children`)
- using the **spread-operator** (`...`) to convert a **NodeList** into an array
- `append()` (use only `appendChild()`)
- `prepend()`
- `replaceWith()`
- `replaceAll()`
- `closest()`
- `replaceChildren()`

If you want to perform these operations, you may use `Array.from()` to first convert the collection into an array.

Requirements

Write a JS program that can load, create, remove, and edit a list of tasks. You will be given an HTML template to which you must bind the needed functionality.

First, you need to install all dependencies using the `npm install` command.

Then, you can start the front-end application with the `npm start` command.

You also must start the ***server.js*** file in the ***server*** folder using the `node server.js` command in another console **(BOTH THE CLIENT AND THE SERVER MUST RUN AT THE SAME TIME)**.

At any point, you can open up another console and run `npm test` to test the **current state** of your application. It's preferable for **all of your tests to pass locally** before you submit to the Judge platform, like this:

```
E2E tests
  Sprint Board Tests
    √ Load Board (loads all in correct categories) (501ms)
    √ Create Task (successful create & clear inputs) (532ms)
    √ Move Task (from ToDo to In Progress) (551ms)
    √ Move Task (from In Progress to Code Review) (504ms)
    √ Move Task (from Code Review to Done) (508ms)
    √ Close Task (497ms)


6 passing (4s)
```

Endpoints

- http://localhost:3030/jsonstore/tasks/
- http://localhost:3030/jsonstore/tasks/:id

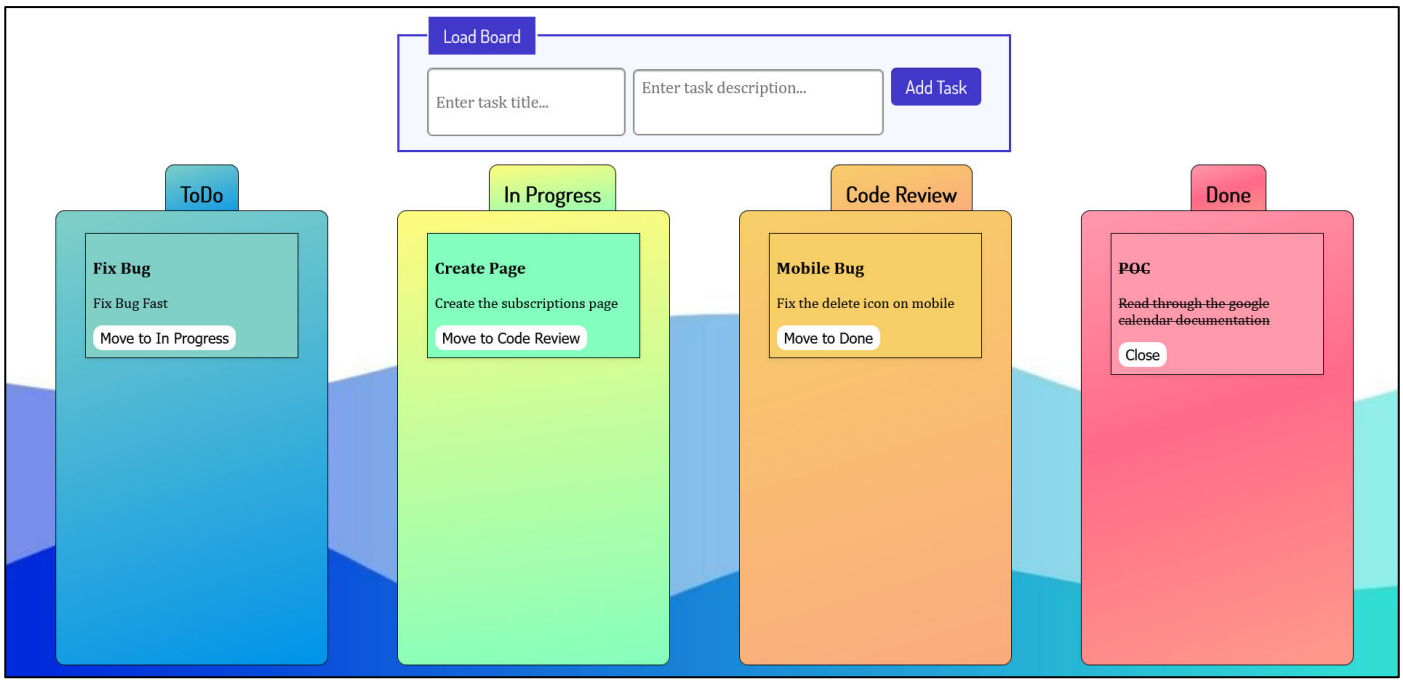
Load the Board

Clicking the **[Load Board]** button on the **top left** should send a **GET** request to the server to fetch **all sprint tasks** in your local database. You have to add each task to its **specified column list** (ToDo, In Progress, Code Review, or Done). The tasks should have **different text contents** inside their respective **buttons** depending on which column they are in:

- ToDo – **"Move to In Progress"**
- In Progress – **"Move to Code Review"**
- Code Review – **"Move to Done"**
- Done – **"Close"**

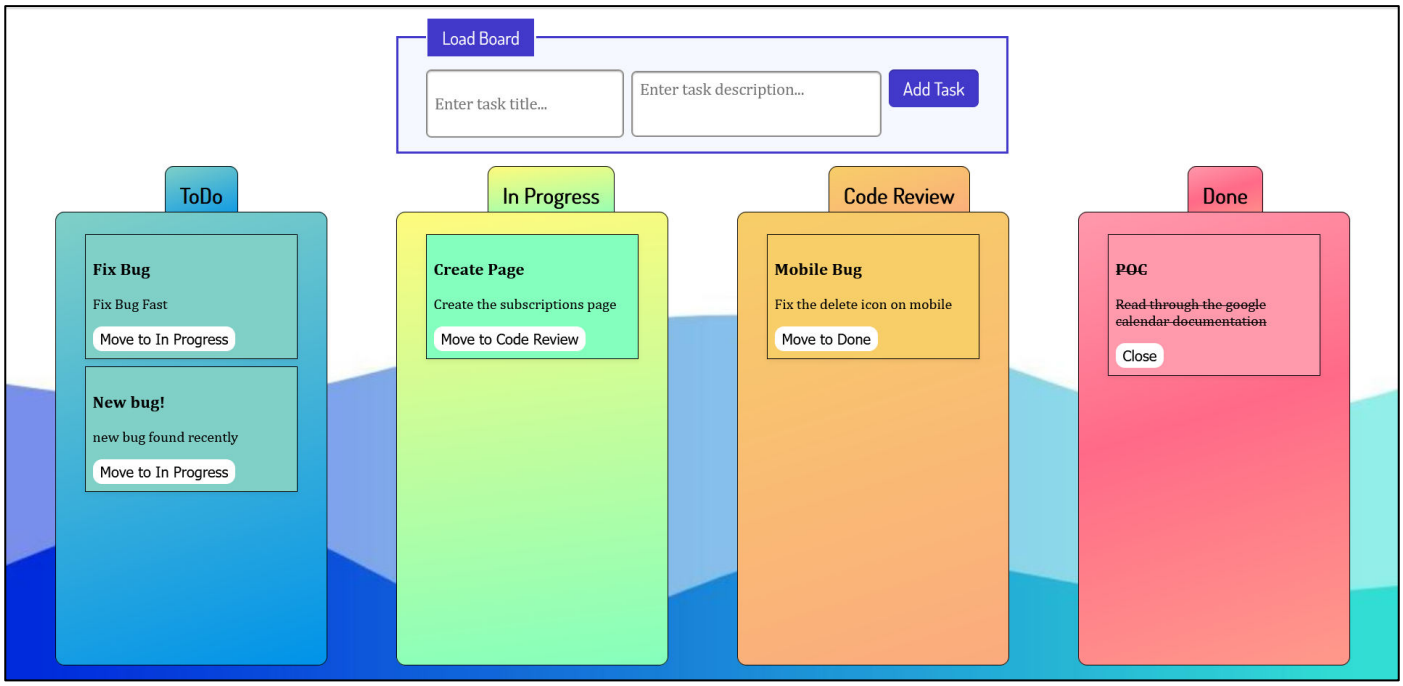
Each Sprint task has the following **HTML structure**:

```
<li class="task">
  <h3>Fix Bug</h3>
  <p>We have a new bug to fix</p>
  <button>Move to In Progress/Move to Code Review/Move to Done/Close</button>
</li>
```



Add a Task

Clicking the **[Add Task]** button should send a **POST** request to the server creating a new task with a **title** and **description** from the input values (the **status** should have an initial value of **'ToDo'**). After a successful creation, you should send another **GET** request to fetch all the tasks, including the **newly added one** into the **ToDo** column. You should also **clear all input fields** after the creation!



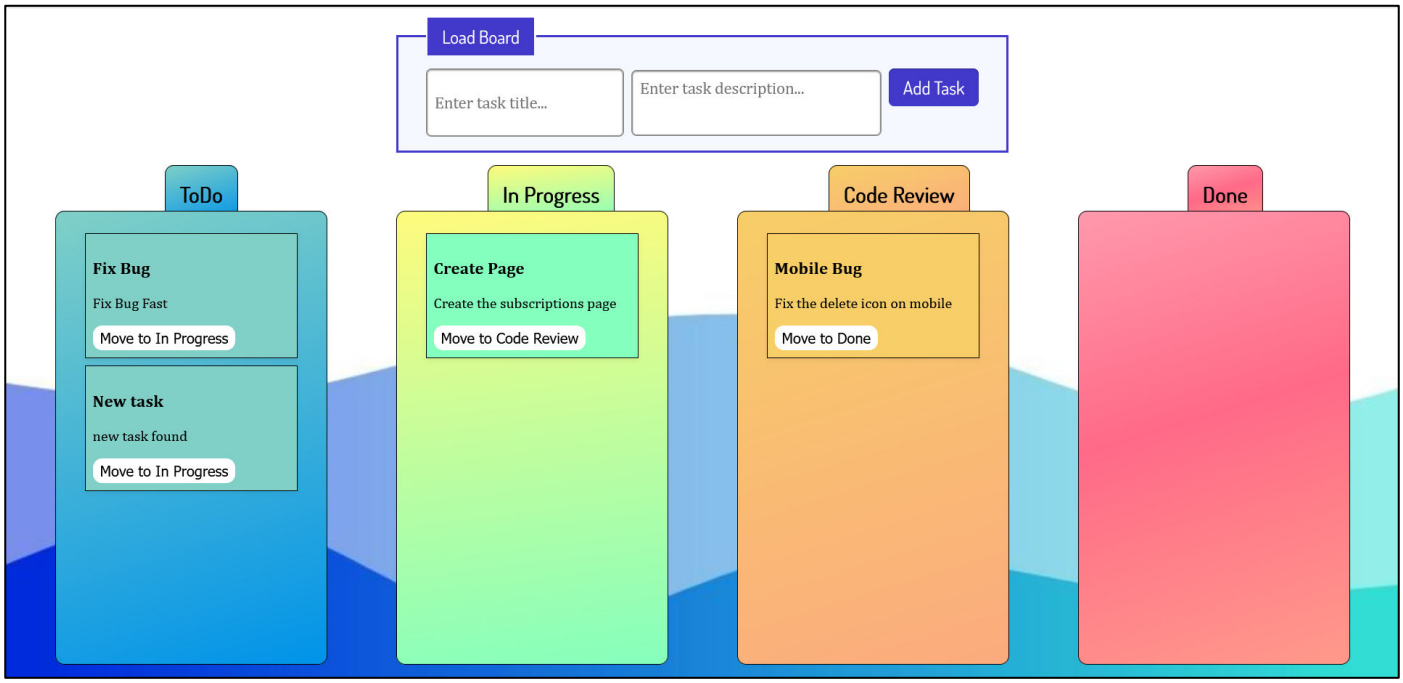
Move a Task

Clicking the **[Move]** button on the individual tasks from the **first 3 columns** should move the task from one column to the next – from **ToDo** to **In Progress** to **Code Review** and finally to **Done**.

After clicking the **[Move]** button, you should send a **PATCH** request to the server to **modify the status** of the changed item. After the successful request, you should **fetch the items again** and see that the change has been made.

Close a Task

Clicking the **[Close]** button on the tasks in the **Done section** should send a **DELETE** request to the server and remove the item from your local database. After you've removed it successfully, **fetch** the items **again**.



Submitting Your Solution

Select the content of your working folder (the given resources). Exclude the *node_modules* & *tests* folders. Archive the rest into a **ZIP** file and upload the archive to Judge.