Problem 3 – Todo List

Problem for exam preparation for the "JS Front-End" course @ SoftUni .

Submit your solutions in the SoftUni judge system at https://judge.softuni.org/Contests/3878/Exam-Preparation-I.

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()` in Judge (use only `appendChild()`)
- `prepend()`
- `replaceWith()`
- `replaceAll()`
- `closest()`
- `replaceChildren()`
- Always turn the collection into a **JS array** (forEach, forOf, et.)

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

Requirements

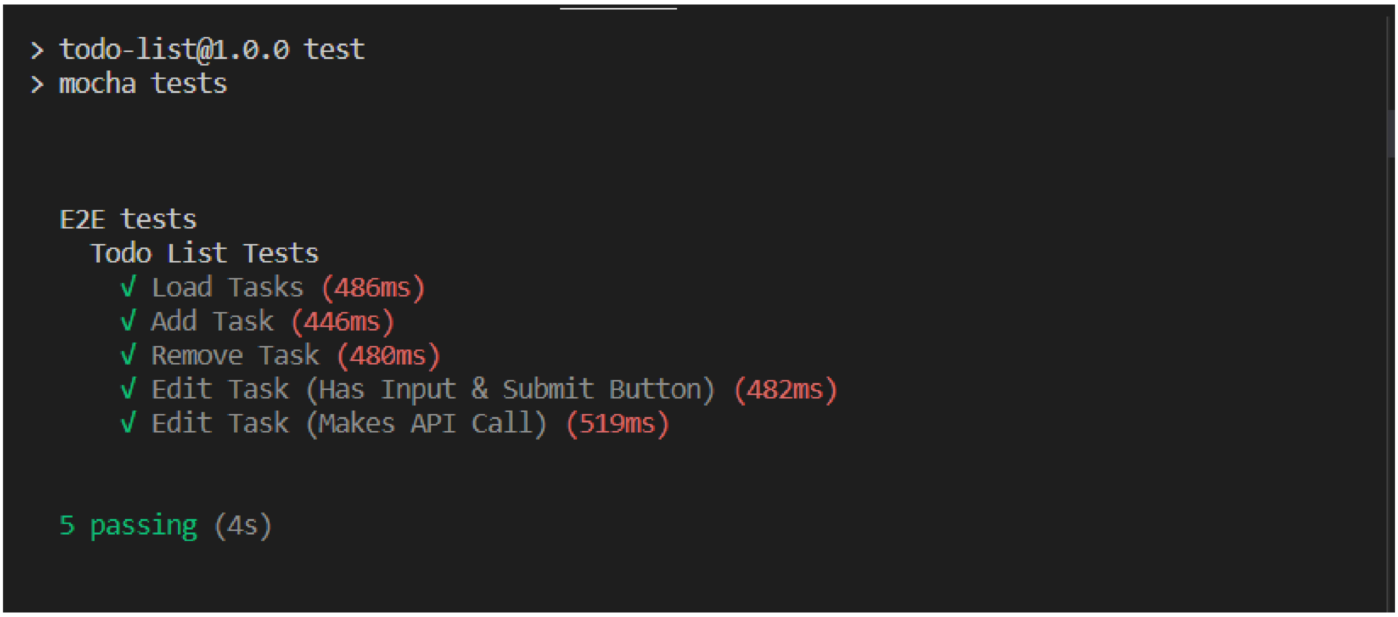
Write a JS program that can load, create, remove and edit a list of tasks. You will be given an HTML template to which you must bind the needed functionality.

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 also 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 test to pass locally** before you submit to the judge platform, like this:



```
> todo-list@1.0.0 test
> mocha tests



  E2E tests
    Todo List Tests
      √ Load Tasks (486ms)
      √ Add Task (446ms)
      √ Remove Task (480ms)
      √ Edit Task (Has Input & Submit Button) (482ms)
      √ Edit Task (Makes API Call) (519ms)


  5 passing (4s)
```

Endpoints

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

Load Tasks

Clicking the **[Load All]** button should send a **GET** request to the server to fetch all tasks in your local database.

Each task is a list item in the following format:

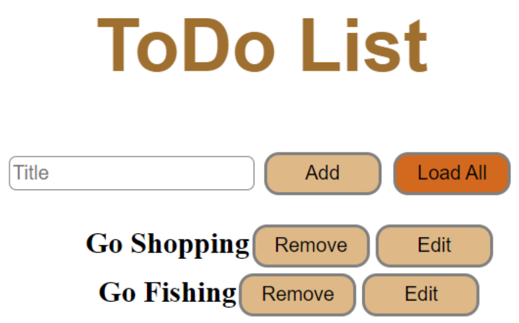
<li>

<span>Go Shopping</span>

<button>Remove</button>

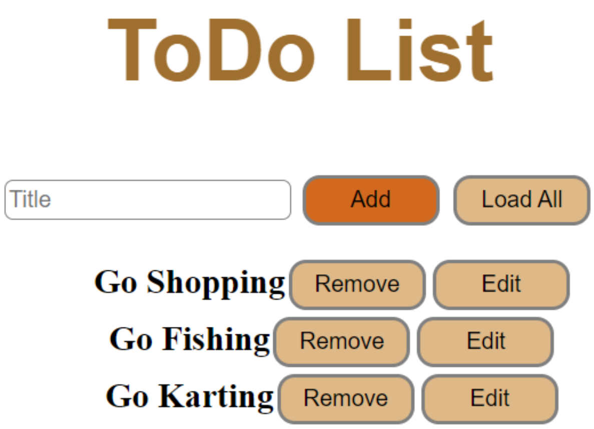
<button>Edit</button>

</li>



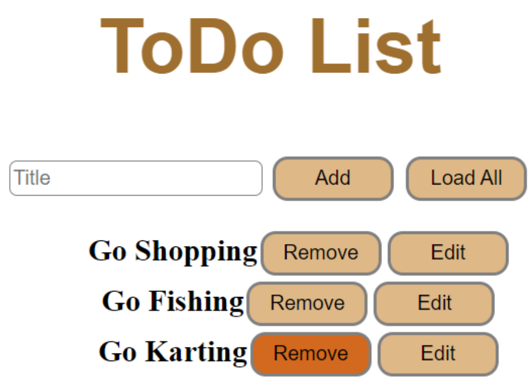
Add Task

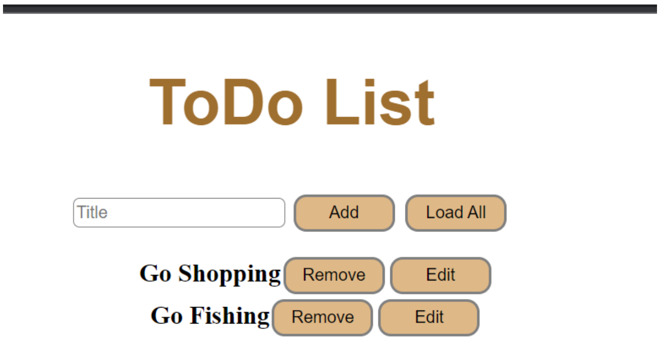
Clicking the **[Add]** button should send a **POST** request to server, creating a new task with the name from the input value and after the successful creation you should send another **GET** request to fetch all tasks, including the newly added one.



Remove Task

Clicking the **[Remove]** button 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**.

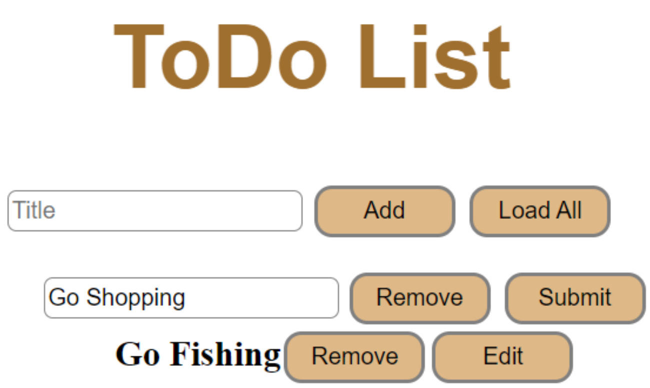




Edit Task

Clicking the **[Edit]** button should change the DOM Structure for that list element, instead of a **span** you should create an **input field** with the **current name as value and exchange the Edit button with a Submit button**.

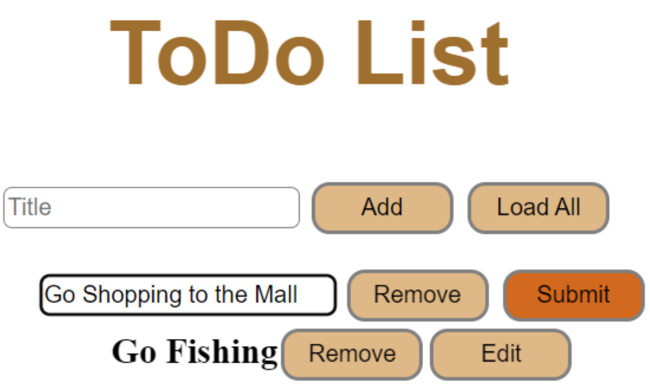
After clicking the [**Submit]** button you should send a **PATCH** request to the server to **modify the name** of the changed item, after the successful request you should **fetch the items again** and see that changes have been made.


ToDo List

Title

Go Shopping

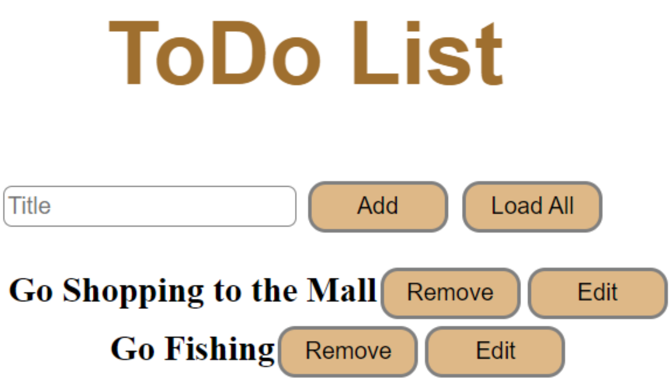
Go Fishing


ToDo List

Title

Go Shopping to the Mall

Go Fishing


ToDo List

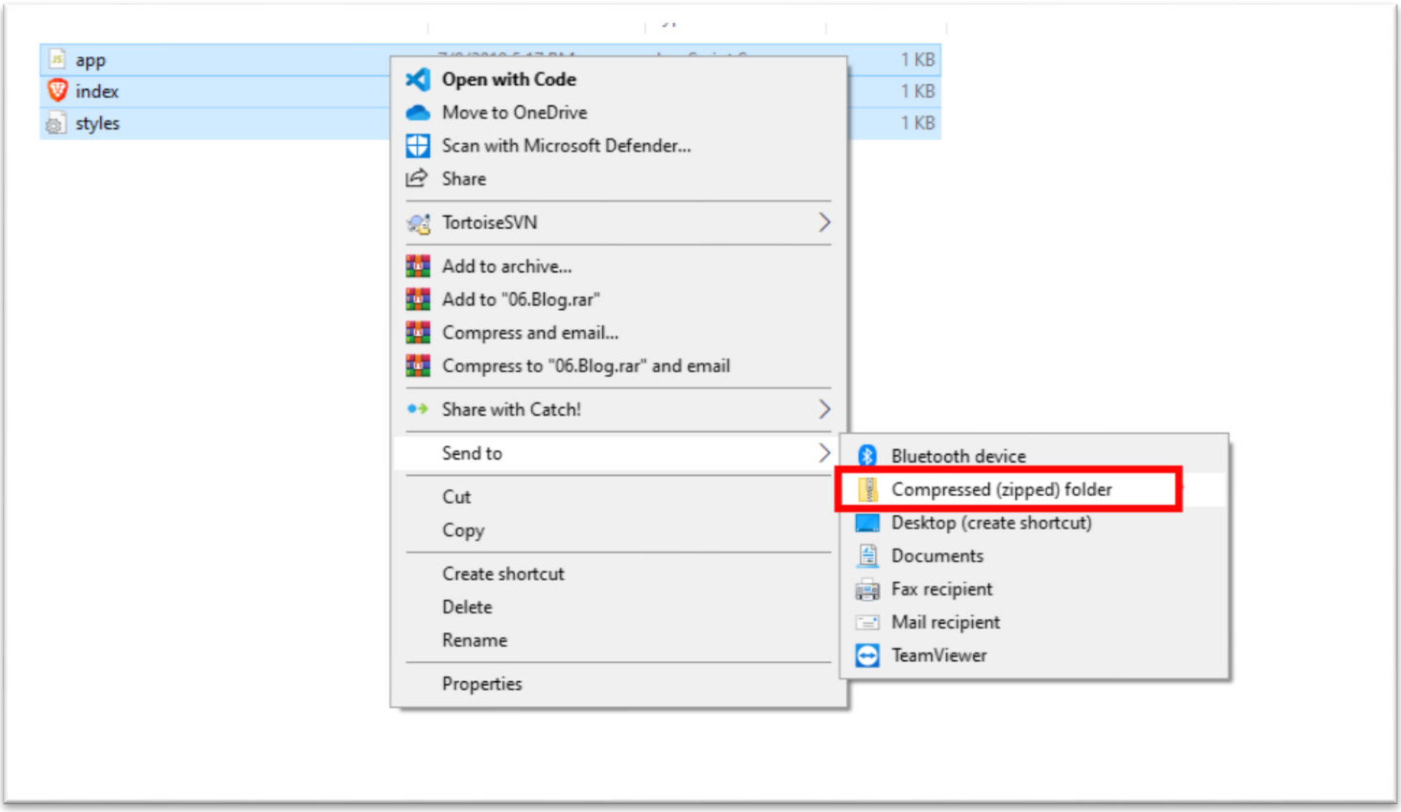
Title

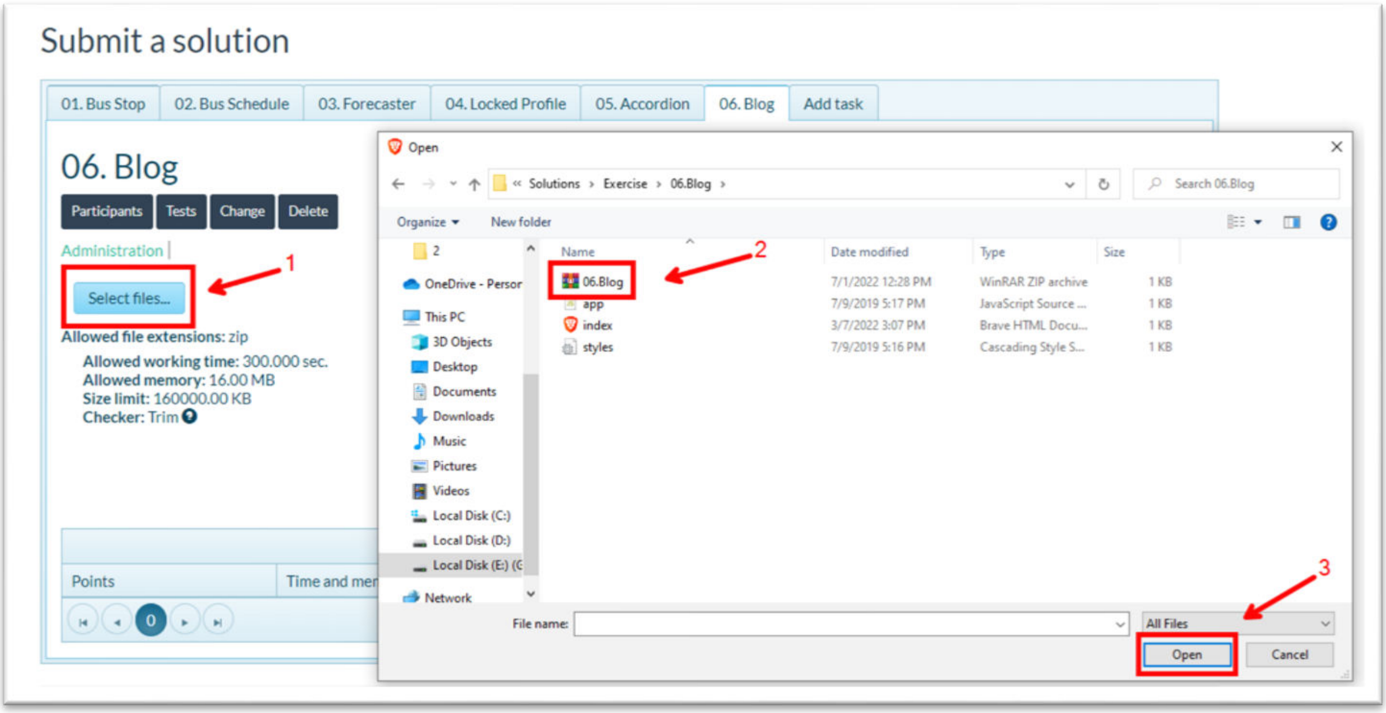
Go Shopping to the Mall

Go Fishing

Submitting Your Solution

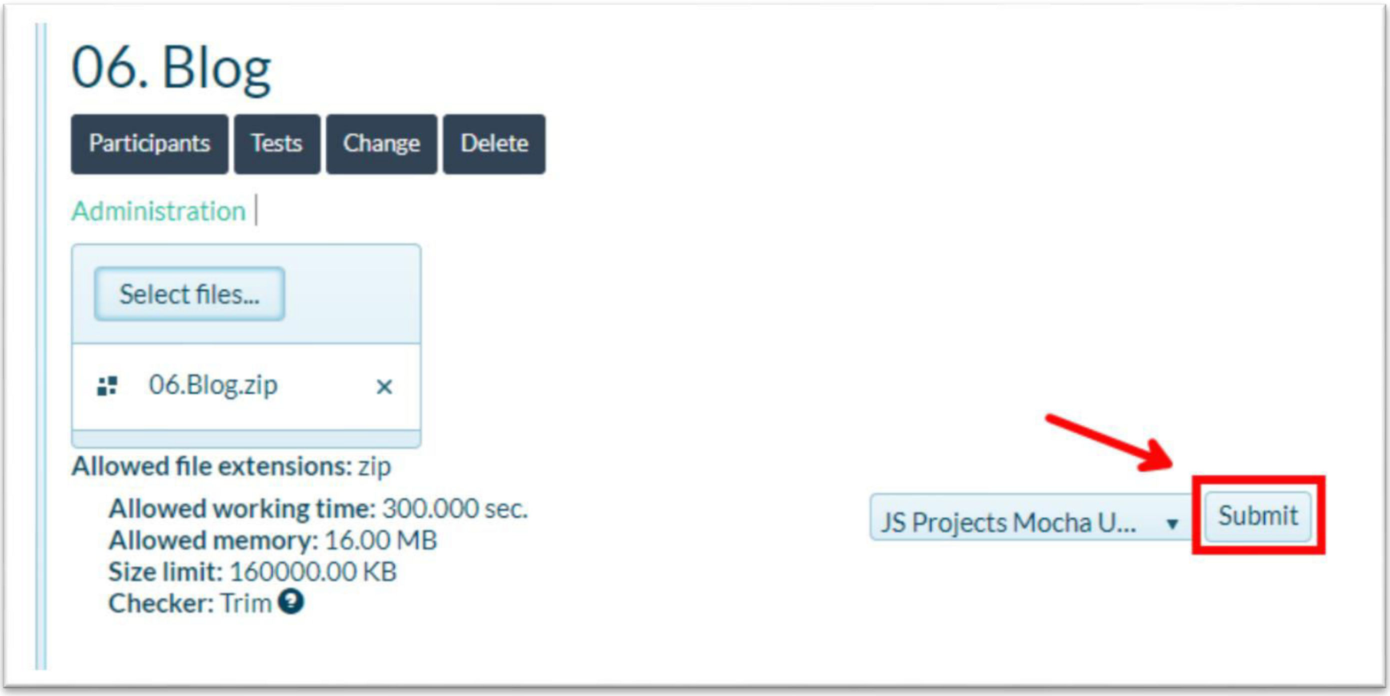
Place in a **ZIP** file the content of the given resources including your solution. Exclude the **`node_modules`** & **`tests`** folders. Upload the archive to Judge.




Submit a solution

06. Blog

Allowed file extensions: zip
Allowed working time: 300.000 sec.
Allowed memory: 16.00 MB
Size limit: 160000.00 KB
Checker: Trim



06. Blog

06.Blog.zip

Allowed file extensions: zip

Allowed working time: 300.000 sec.
Allowed memory: 16.00 MB
Size limit: 160000.00 KB
Checker: Trim

JS Projects Mocha U...