Project Setup for Frontend after Backend

React setup:

npx create-react-app frontend

clean up: delete all, just keep App.js, index.js, and index.css, then delete some unused code

Create in ‘scripts’ in package.json (outside) to run both backend + frontend:

npm i axios react-icons react-toastify node-sass

npm i concurrently -D (for each project, or -g for all)

"frontend": "npm start --prefix frontend",

"both": "concurrently \"npm run backend\" \"npm run frontend\""

Check: npm run both

Create assets (for pictutes,..) + components (for coding files js,…) folders: Make them easy to keep track and find.

Snippet: rafce

Import to use react-toastify:

open npmjs.com → react-toastify → copy import code → in app.js:

```
import { ToastContainer } from 'react-toastify';
import 'react-toastify/dist/ReactToastify.css';
import TaskList from './components/TaskList'; // Depend on  file in components folder

function App() {
  return (
    <div className="app">
      <div className="task-container">
        <TaskList /> // This is a file in components folder
      </div>
      <ToastContainer />
    </div>
  );
}

export default App;
```

Import icons from react-icons:

```
import {FaCheckDouble, FaEdit, FaRegTrashAlt} from 'react-icons/fa';
```

```
<FaCheckDouble color = "green" />
```

Noted Demo:

```
import { useEffect, useState } from "react";
import { toast } from "react-toastify";
import axios from "axios";
import { URL } from "../App";



const createTask = async (e) => {
    e.preventDefault();
    if (name === "") {
    return toast.error("Input field cannot be empty"); //from react-toastify
    }

    try {
     await axios.post(`${URL}/api/tasks`, formData); //Scroll down, set PROXY
       toast.success("Task created successfully");
       setFormData({ ...formData, name: "" });
```

```
    } catch (error) {
        toast.error(error.message);
    }
};
```

Set up cors(): allow send data from frontend to backend

in server.js in backend folder:

npm i cors

```
const cors = require('cors');

app.use(cors({
    origin: [“http://localhost:3000/”]
}));
```

Check: localhost + Insomia

Set PROXY URL:

await axios.post**(`${URL}/api/tasks`**, formData); **//Scroll up for more information**

in frontend folder, package.json:

"name": "frontend",

"version": "0.1.0",

"private": true,

"proxy": "http://localhost:5000",

"dependencies": {…}

in app.js:

export const URL = process.env.**REACT_APP_SERVER_URL;**

→ Create REACT_APP_SERVER_URL:

in frontend folder, create .env file:

REACT_APP_SERVER_URL=http://localhost:5000

Set 'loading' image:

in the main js file:

import loadingImg from "../assets/loader.gif";

const [isLoading, setIsLoading] = useState(false);

```
const getTasks = async () => {
  setIsLoading(true);
  try {
    const {data} = await axios.get(`${URL}/api/tasks`);
    setTasks(data);
    setIsLoading(false);
  } catch (error) {
    toast.error(error.message);
    setIsLoading(false);
  }
};
```

in return() of that main js file:

```
{
    isLoading && (
      <div className="--flex-center">
        <img src={loadingImg} alt="Loading" />
      </div>
    )
    }

 {
    !isLoading && tasks.length === 0 ? (
```

```
            <p className="--py">
              No tasks to display. Create a task.
            </p>
          ) : (
            <>
            {tasks.map((task, index) => {
              return <Task
              key={task._id}
              task={task}
              index={index}
              deleteTask={deleteTask}
              getSingleTask={getSingleTask}
              setToComplete={setToComplete}

              />;
            })}
            </>
          )}
```

Buton change when updating:

```
<button type="submit">{isEditing ? "Edit" : "Add"}</button>
```