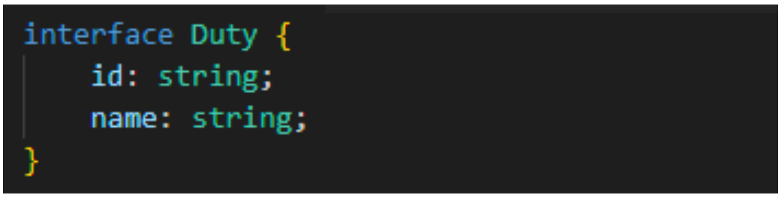
TECHNICAL TEST

This technical test aims to create an end-to-end web application that allows the user to read, create and update a to-do list of duties of any kind.

An example of the data structure for each duty is as follows:



```
interface Duty {
    id: string;
    name: string;
}
```

General requirements

- It must be able to perform read, create and update operations.
- The code is published in a public git repository where anybody can clone the solution.
- A README.md file is written with clear instructions on how to run the solution. In the case of the frontend project, provide screenshots.
- Build and serve commands must run on all operating systems (Windows, macOS and Linux).
- Production-ready software, meaning readability and observability.
- Design the structure, architecture, and layers to grow the codebase and become a large system with more functionality.
- Correct error communication and error handling.
- Testing using Jest (frontend & backend).
- Omit any user authentication; the TODO list will be visible to any user.

Frontend requirements

- A frontend developed in React written in TypeScript in strict mode, using hooks that render the list of duties retrieved from the backend and allow the user to create new ones and/or modify existing ones.
- Must include form validations.
- The frontend project must be independent of the backend project.
- Do not use server-side implementation in the frontend, only client-side implementation; consequently, avoid solutions like Next.js or similar.
- Avoid any state management solution like Redux, the React's hook useReducer or similar.

Backend requirements

- A backend written in Node JS with TypeScript in strict mode.
- The data must be stored using PostgreSQL as a database engine.
- Do not use ORM or any other solution that hides the fact you are using a SQL database; use plain SQL queries instead.
- For the purpose of this technical assessment, do not use complex frameworks in the backend, no Nest.js or similar.

Optional requirements

- The solution is also able to delete items from the to-do list.
- Ant Design as the component library for the frontend.