Coding Challenge

The goal of the task is to implement what could be a To Do application. The task consists of two parts, front end and back end.

- Source code should be provided in GIT repository (Github , Bitbucket, Gitlab or other platform). Do commit early and often.
- Cover your code with tests;
- Use best practices and coding conventions;
- Provide documentation how to build and run your solution;

TASK 1

Back-End

Using technologies of your choice (for example: *Spring Boot*), implement back end with the API which would provide an API user with the following capabilities:

- Create A Todo;
- List All Todo items;

Front-End

Using technology of your choice (*React, Vue.js or Angular*) implement a UI for you Back End application, which has the following capabilities:

- Provides a button to add new Todo item;
- Lists all the Todo items;

TASK 2

- Mark the Todo item as archived, so it moves to the archived list;
- Show all archived Todo items;

TASK 3

- Use MySQL or PostgreSQL as a database;
- Avoid using **Hibernate** (use **jDBI** or **jOOQ**);
- Use **liquibase** or **flyway** for DB scripts;

TASK 4

- Use **React** + **MobX** or **Redux** for state management;
- Use **React Router** for routing between components;
- For the UI look & feel make use of **Material-UI** library;

TASK 5

- Provide script in *package.json* file to build docker image;
- Provide a **Gradle** task to build docker image;

Hint: In order to "*dockerize*" your UI project, you might find this helpful: Dockerizing React Application

An example of how UI might look like, the important part is not the looks, but to get both parts (front end and back end connected).