Developer Test

Please see the following user stories below, outlining the business requirements for a simple **to-do** list application.

General Guidelines:

- **Read each story carefully before start to implement it.** Try to follow the incremental approach and finish each story before start implementing the next one.
- Think of the software design you want to apply. **Keep it simple, but do not leave out any of the good practices you know.**
- We recommend to timebox the test to 2-4 hours. **The goal is not to complete all the requirements.**
- **Clean code is more important than implementing all the user stories.**
- Keep the use of external packages to a minimum.

Technical requirements:

- Backend:
    - You must develop a RESTful API using .NET Core or Node.
    - **Your code should have appropriate test coverage.**
    - You don't need to persist any information if you think it is not necessary. Keeping all information in memory, even if it's lost after restarting the application, is an acceptable approach.

- Frontend:
    - You must implement a SPA using the JavaScript framework or UI library that you feel more comfortable using. (React, AngularJS, Vue.JS, ...).

Nice to have:

- Frontend:
    - Use Typescript.
    - Use React as JavaScript UI library.
    - Use a JavaScript testing framework (such as Jest).

- Use a local git repository with frequent commits in order to see a little of your working methods.

Expected output:

Your solution should be added to a public repo (e.g., Github or similar) The solution should contain all your source code, plus a markdown file answering the following questions:

- How long did you spend on your solution?
- How do you build and run your solution?
- What technical and functional assumptions did you make when implementing your solution?
- Briefly explain your technical design and why do you think is the best approach to this problem.
- If you were unable to complete any user stories, outline why and how would you have liked to implement them.

User Stories:

User story 1

As a user, I want to be able to see my list of to-dos, so I can easily see my completed and pending tasks.

Acceptance criteria

1. It should be possible to see all tasks:
    a. Each task will be represented by a simple text description.
    b. Two main tasks groups: pending and completed tasks.
        i. Depending on the type (completed vs pending) the task will be displayed on a different group.

2. Initially, this list will be empty.

User story 2

As a user, I want to be able to add new tasks, so I can have new to-dos in my list.

Acceptance criteria

1. The user should be able to enter a task description.
2. The user should be able to add this previous description to his or her to-do list
3. The added to-do will be displayed as a pending task

User story 3

As a user, I want to be able to change the status of existing tasks, so I can complete it or mark it as pending.

Acceptance criteria

1. The user should be able to pick a task and change its status:

    - If the task is pending it will become completed
    - If the task is completed it will become pending

2. The to-do list should be updated accordingly, moving tasks between groups.