React Micro Frontend Todo List

Objective

The goal of this exercise is to develop a standalone React Micro Frontend (MFE) component that encapsulates a fully functional todo list application. The idea is to evaluate your ability to design and structure a maintainable React application, while paying close attention to code organization, testing, and TypeScript usage.

Time Expectation

This exercise should not take more than 8 hours to complete.

Requirements

Your React MFE component should deliver the following features:

- Todo Creation:
  - Users can input a new todo task description.
  - Tasks are added to a list.
- Todo Status:
  - Tasks should have a checkbox to mark them as completed or incomplete.
  - Visually distinguish between completed and incomplete tasks (e.g., strikethrough on completed items.)
- Todo Persistence:
  - Todo items should be saved using the browser's localStorage to persist across page refreshes and sessions.
- Bonus Points:
  - Provide buttons or a mechanism to filter the list: "All", "Active", "Completed".

Technical Considerations

- TypeScript: The project should be written in TypeScript for type safety and maintainability.
- Testing: Include unit tests with meaningful coverage for core component logic.
- Code Quality: Follow best practices for clean, readable, and well-structured code.
- State Management: React's built-in state management is acceptable for this exercise. Overly complex solutions are unnecessary.

Gotchas

● Focus on the MFE aspect: Ensure the component can be seamlessly integrated into various host applications. Consider how you'd handle potential prop/data communication.

● Edge Cases: Think about error handling and edge cases. What happens if localStorage is unavailable, or there's invalid input?

Submission & Evaluation

● Packaging: Provide a clear way to run your MFE (e.g., instructions if a simple setup is required.) Consider if it would make sense to package your MFE.

● Documentation: Include a short README with setup instructions and an overview of your thought process on design and architectural choices.

Evaluation Criteria

● Functionality: Does the component meet all the specified requirements?

● Code Quality: Is the code well-written, maintainable, and adheres to TypeScript best practices?

● Testing: Does the component have thorough and meaningful tests?

● Micro Frontend Considerations: How well does the solution demonstrate an understanding of micro frontend principles and potential integration challenges?

● Attention to Detail: Does the submission address edge cases and show careful thought?

Additional Notes

● You are not expected to spend time on elaborate styling. Basic, functional styling is sufficient.

● Feel free to make reasoned decisions about design patterns and architectural choices. Be prepared to discuss these during the interview.