Task: Todo App

Build a responsive Todo application that demonstrates solid state management, clean UI, and basic statistics.

Objectives

- Implement a Todo app with add, edit, toggle complete, delete, and filter (All/Active/Completed).
- Manage global state using React Context or Redux.
- Display live stats in the header (e.g., total todos, completed, remaining), driven by the global state.
- In the completed state there will be no edit and checkbox.
- Create a clean, responsive UI.

Requirements

- Framework / Library: React, (TypeScript preferred) but optional, CSS framework is your choice.
- Global State:
    - Store todos globally (Context or Redux).
    - Actions: addTodo, toggleTodo, editTodo, deleteTodo, clearCompleted, setFilter.
    - Derived selectors for stats (total, completed, remaining).
- UI/UX:
    - Header with app title and live stats.
    - Input to add a new todo in form modal.
    - Todo list with:
        - Checkbox to toggle complete
        - Inline edit or edit modal
        - Delete control
        - Loading state
    - Filters: All, Active, Completed
    - “Clear completed” button
    - Empty states and basic validations (e.g., prevent empty titles).
    - Responsive layout (mobile, tablet, desktop).
- Nice-to-have:
    - Persistence with localStorage.
    - Keyboard accessibility (Enter to add, Esc to cancel edit).
    - Animations for add/remove/toggle.
    - Dark mode toggle.
    - Redux Toolkit + RTK selectors or Context + useReducer with memoized selectors.

What to Deliver

- Public repo or zip with:
    - Source code
    - README with setup/run instructions, brief architecture notes.
- Deployed preview link (Netlify/Vercel/GitHub Pages) is a plus. (Encouraged)

Note: The use of AI assisted code editors for completing this task is discouraged. Any indication of such usage may invalidate the assessment. While we welcome the use of AI tools, we also want to ensure that candidates can demonstrate their own understanding and problem-solving skills.