A screen shot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screen shot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

Todo List Application

##Input

The application accepts user input through:

* A text input field where users can type the description of their task
* A button click interaction to add the task to the list
* Checkbox interaction to mark tasks as completed or incomplete
* Another button click interaction to delete specific tasks from the list
* Keyboard interaction (pressing Enter) to add a task as an alternative to clicking the button

These inputs represent the primary way users interact with the application to manage their todo items.

##Process

The application processes user interactions using React's state management with the useState hook:

* When a user types in the input field, the inputValue state is updated with each keystroke
* When the "Add Task" button is clicked or Enter is pressed, the application validates that the input is not empty
* If valid, a new todo object is created with a unique ID (using the current timestamp), the entered text, and a default completed status of false
* The new todo is added to the todos array state using the spread operator to maintain immutability
* When a task's checkbox is clicked, the application maps through the todos array and toggles the completed status of the matching item
* The application tracks and displays statistics about completed versus total tasks
* When a "Delete" button is clicked for a specific todo item, the application filters the todos array to remove the item with the matching ID

The component structure separates concerns:

* App.js serves as the main container
* TodoList.js manages state and provides the input field and container for todo items
* TodoItem.js renders individual todo items and handles delete functionality

##Output

The application outputs a visual representation of:

* The current input value in the text field
* The list of all todo items, each with its description, a completion checkbox, and a delete button
* Visual indicators for completed tasks (strikethrough text and grayed-out appearance)
* A statistics section showing the number of completed tasks out of the total
* A message when the list is empty to provide user feedback
* Visual feedback through CSS styling to make the interface intuitive

The todo list updates dynamically as users add, complete, or remove items, providing immediate feedback for their actions. The clean interface with contrasting colors for add and delete buttons and visual differentiation for completed tasks makes the application easy to use and understand.