Core | Deadline: Sunday of Week 3 | Difficulty Level: Advanced | Est. Time: 00:00-04:00

Todo List (Core)

Put together everything you've learned to create a dynamic task tracking application in React!

Learning Objectives

- Demonstrate proficiency in React concepts such as props, state, lifted state, component hierarchy, map and filter.

Welcome to another **Core assignment**! Some students like to explore the assignments before they're finished reading through the lessons, and that's okay! It can be good for your brain to have a preview of what your future challenges might be. However, before you begin this assignment, it's important that you've first:

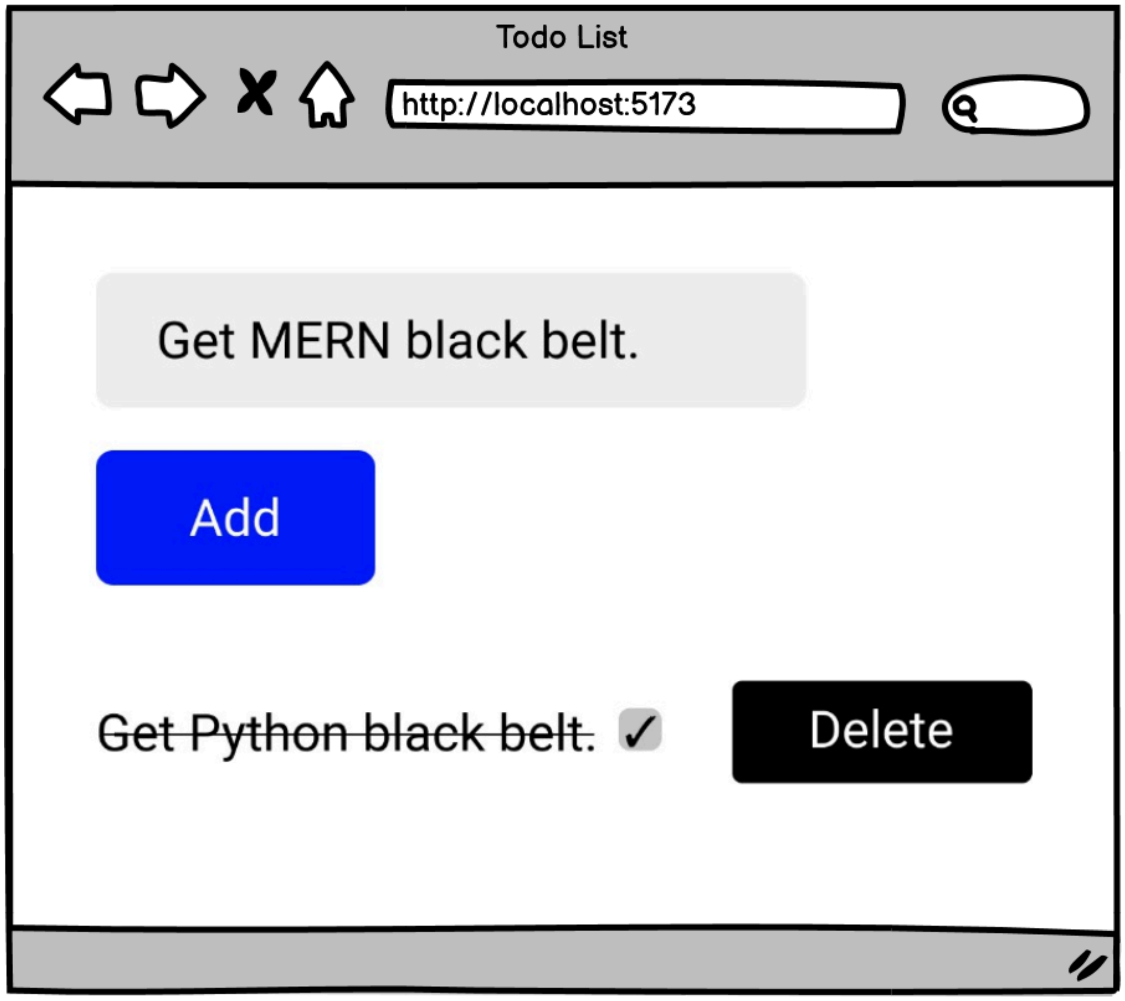
- Completed the preceding lesson modules
- Taken the knowledge checks to confirm your understanding
- Viewed lecture material related to the assignment topics
- Completed and submitted your practice assignments

Todo List:

Ah, the todo list! A ubiquitous project for any respectable JavaScript framework. Your job is to create a small React application to store and modify a list of tasks. Each task will have a text string as well as a completed property, which will be set to false initially. As you check off items, they should appear slashed out on the page. Using what you know about state and iterating through lists, render a list of items, and give the user the option to remove each item and add new ones. There are different ways to implement the remove function. One way to do it is using the built-in filter method. Filter returns a new array when invoked and is a great way to stay true to the functional nature of React.

Hint: When calling `setState`, make sure to send in a brand new object or array.

Your component structure should include at least three components with a state holding the array of "todo" items lifted to the common parent component.



For an even greater challenge, once you've completed all of your React unit core assignments, become a master with Super Todos. Super Todos will ask us to refactor to utilize the presentational-container design pattern and the useReducer hook. Both of these topics will be mastery items on the end-of-course Belt Exam. Learn more about these topics and tackle Super Todos in the React Master section at the end of this unit.

- [ ] Iterate through the existing tasks using the map method.
- [ ] Allow the user to add a new task, initially set to not completed. (Hint: The task should be added to the array as one field in an object.)
- [ ] The completion value should be a property in the new task object - not a standalone stateful value
- [ ] Allow the user to remove a task by clicking the delete button.
- [ ] Allow the user to toggle a task's completion property by clicking the checkbox next to it.
- [ ] When updating the tasks' state, don't mutate the current task objects or the current task list. (Hint: Use map to selectively return copies of array elements)