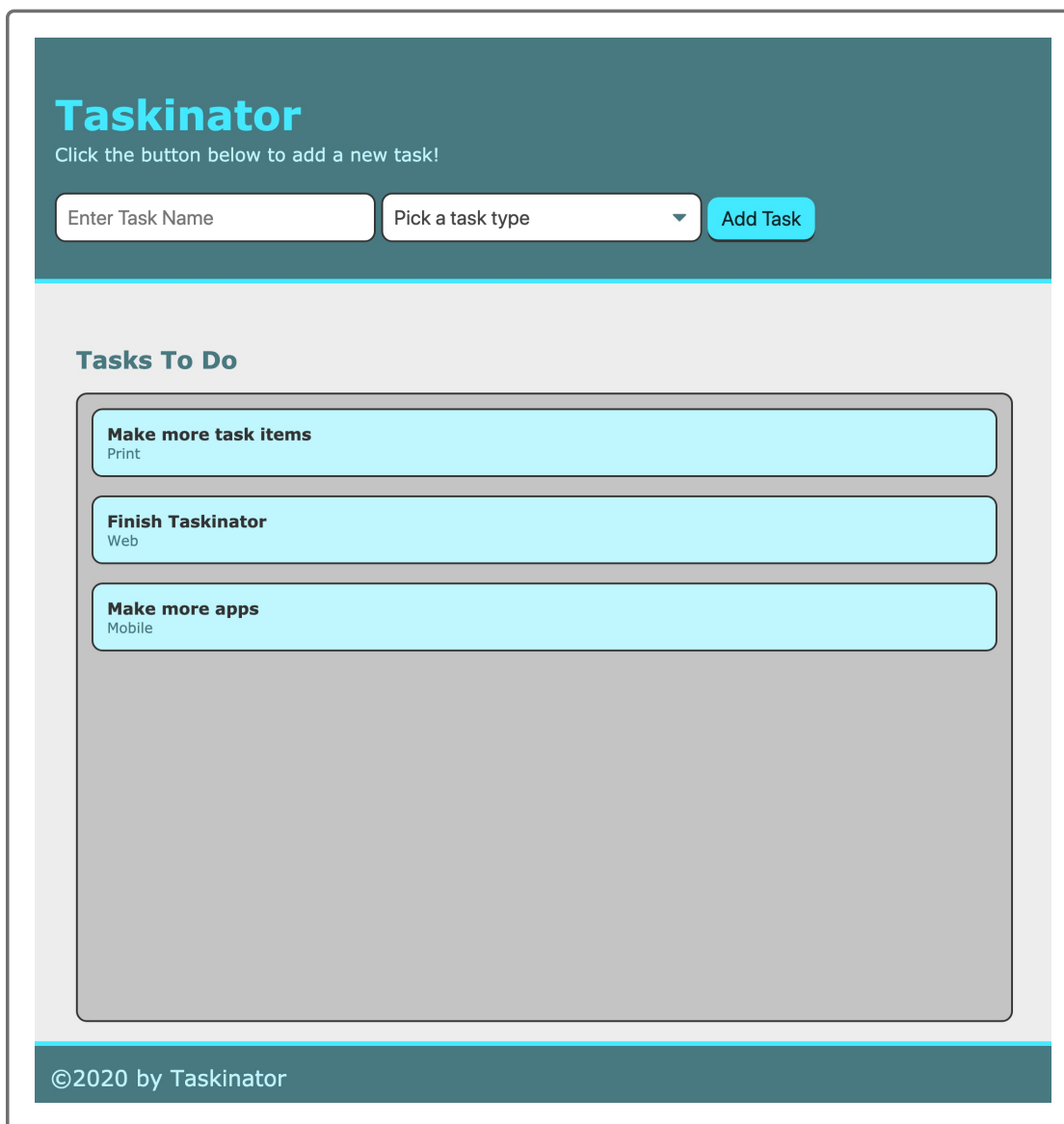


4.2.2 Preview

By the end of this lesson, we'll build a form that will allow the application to create new tasks based on user input.

Here's what the form should look like:



The image shows a web application mockup for 'Taskinator'. The header is a dark teal bar with the title 'Taskinator' in large, bold, light blue font. Below the title, a smaller line of text says 'Click the button below to add a new task!'. The main form area is white and contains three elements: a text input field labeled 'Enter Task Name', a dropdown menu labeled 'Pick a task type' with a downward arrow, and a blue 'Add Task' button. Below the form, there is a section titled 'Tasks To Do' in a dark teal font. This section contains a list of three tasks, each in a light blue box with a dark teal border. The first task is 'Make more task items' with a sub-label 'Print'. The second is 'Finish Taskinator' with a sub-label 'Web'. The third is 'Make more apps' with a sub-label 'Mobile'. Below the list is a large, empty gray rectangular area. At the bottom of the page, a dark teal footer bar contains the text '©2020 by Taskinator' in white.

The form will have the following features:

- The user will be able to create a task by filling out a form with the task name and choosing what type of task it is.
- The user will be able to submit a task by pressing either the Add Task button or the Enter key on the keyboard.
- If either form field is empty, the task won't be created.

There aren't many features to add here, but they are features seen and used throughout modern web development.

Like in the last lesson, adding features in a certain order boosts efficiency. Let's outline the steps we'll take:

1. Create a new Git branch. We'll create a new feature branch where we will code the form.
2. Add task form to HTML. We'll add an HTML form to the Taskinator, which will allow the user to enter the task name and type.
3. Handle form submission. We'll use JavaScript to add a task to the list when the "Add Task" button is clicked.
4. Capture form field values. We'll use JavaScript to capture the unique information the user enters (the task name and type).
5. Organize functionality. We'll refactor the code to make it more maintainable.
6. Address usability concerns. We'll improve the user experience by validating form input and resetting the form after the user clicks the "Add Task" button.
7. Save our progress with Git. We'll commit and push our changes up to GitHub for safe-keeping.

Now we have a high-level idea of what each step will entail, even if we don't know all the details just yet. Let's start with the usual and create the new Git feature branch!