5.2.5 Add Bootstrap Components

The task lists in the client's mock-up look a bit different from what we have, so we'll have to consider how to achieve that design. Again, we could create custom CSS for all of these styles, or we could see if Bootstrap has anything for us. Luckily, they do, and we can find it under the Components tab on the left side of their documentation.

ON THE JOB

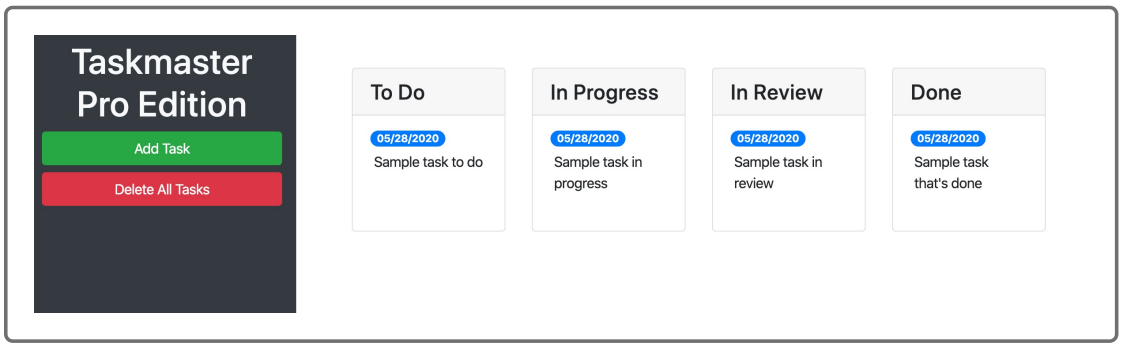
UI components have become a big part of web development over the years. Instead of thinking about the whole page as we build it, we can think about it one piece at a time.

If we want to build a form, we can think about how the form looks in general rather than how the form looks in that particular spot on the page. This way, if we need to move the form or use it again, we can put it in a different spot on the page and it will still look nice.

Browse the different components Bootstrap offers and compare each one to the mock-up from the client. We're looking for a **card** component. Implementing cards will help the task lists stand out from each other, so let's add them!

Build Cards

For background information, see **the Bootstrap documentation for cards (https://getbootstrap.com/docs/4.3/components/card/)** . We'll begin to work them into the layout to look like this image:



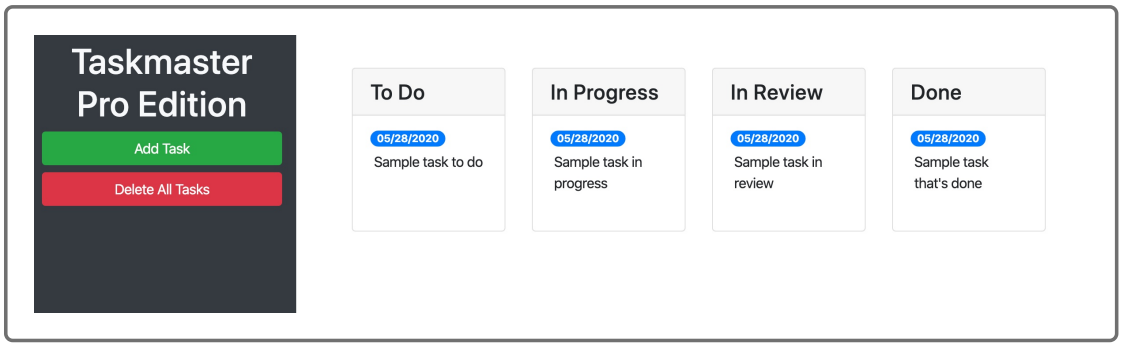
Cards encapsulate related content. You can use them for news articles, products in an online store, or in this case, organizing lists by status. Let's turn the list-status columns into cards and make the HTML in each column look like this:

```
<div class="col-12 col-md-6 col-xl-3 mb-3">
  <div class="card">
    <h4 class="card-header">
      To Do
    </h4>
    <ul id="list-toDo" class="list-group list-group-flush">
      <li class="list-group-item">
        <span class="badge badge-primary badge-pill">05/28/2020</span>
        <p class="m-1">Sample task to do</p>
      </li>
    </ul>
  </div>
</div>
```

IMPORTANT

> Do this for all of the task-list columns, but don't just copy and paste the preceding code. Each list needs a unique title and `id` attribute!

Save the code and then open the app. Use the Delete All Tasks button to delete all previous tasks, and then refresh the page. It should look like this image:



This doesn't match the color scheme the client showed us, but we'll get to that shortly, as it will be easy to update with Bootstrap utility classes. For now, let's focus on the card component we just implemented.

The documentation for cards tells us there are many variations. Some feature headers and footers, some use buttons, and some have images; their use cases vary. We'll use a card with a header and a list inside it.

DEEP DIVE

DEEP DIVE

For more information about the type of card we created, read **Bootstrap's documentation on list groups in cards (https://getbootstrap.com/docs/4.3/components/card/#list-groups)** .

Now that we know what components are and how they work, we can probably implement them in a few other areas to enhance the UI.

Modals

The client gave us their app with a few Bootstrap components already in place. Some of them are simple, such as the buttons in the left column. Others are a little more involved, like the modal window that pops up when we click the Add Task button.

So how does this work? Bootstrap relies on jQuery and another library, called Popper, for some of its features, like opening a modal. That's why at the bottom of the `index.html` file, the Bootstrap CDN's JavaScript file loads after the others. It requires them to load before it loads its own code.

To learn more about using them in different situations, read the **Bootstrap documentation on modals (https://getbootstrap.com/docs/4.3/components/modal/)** . The most important thing to know is how to trigger a modal to open with a button. In our code, the HTML towards the bottom of the file for the modal has an `id` attribute set to `task-form-modal`. And the button that we can click to open the modal has a custom data attribute called `data-target` set to `#task-form-modal`. Notice the similarity?

Bootstrap uses a custom data attribute to connect a button to the modal that it opens. If the value of the `id` for the modal changes, then the `data-trigger` attribute for the button needs to match it in order to work.

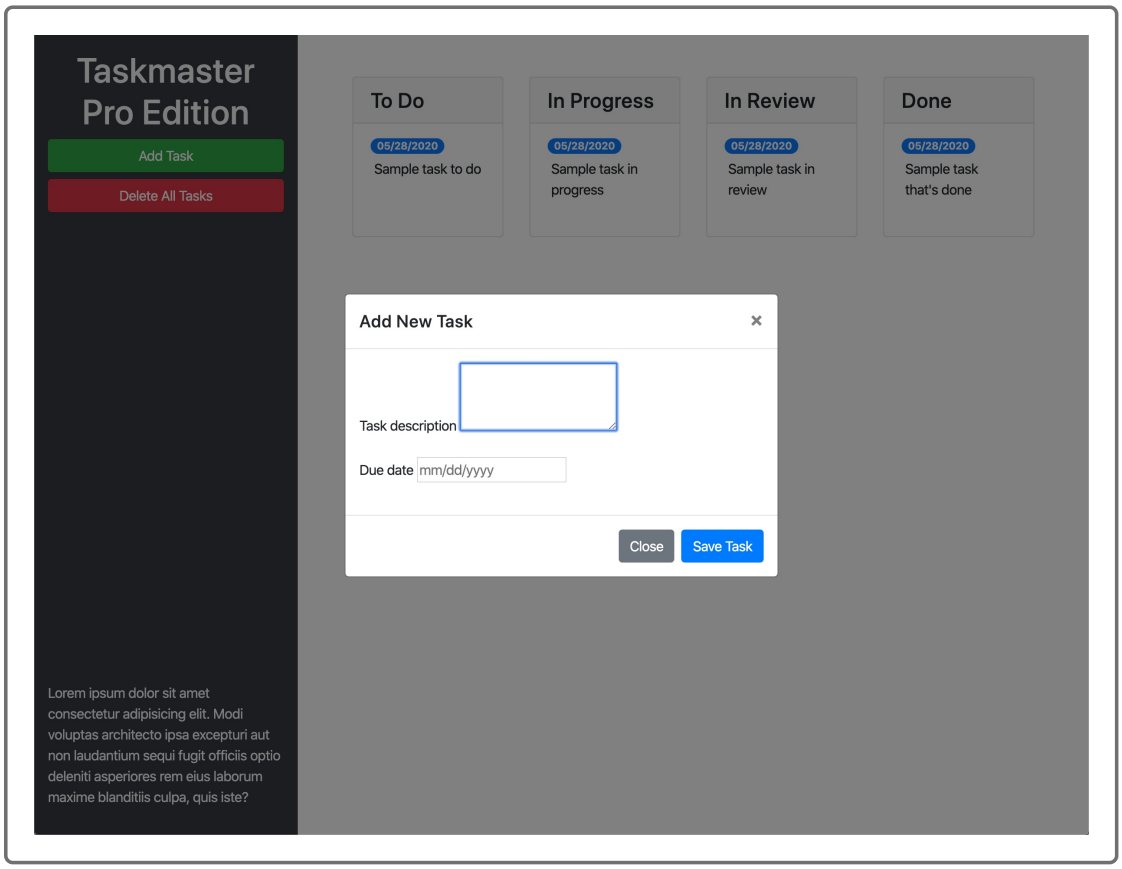
IMPORTANT

> We wouldn't know how to use components like these without the documentation. That's why it is vital to read through documentation

> whenever possible!

Forms

The form looks a little unpolished when we open the modal, like this image shows:



Luckily, we don't have to worry about our own styles. We can just use the classes that **Bootstrap's form component (https://getbootstrap.com/docs/4.3/components/forms/)** gives us!

Like the varied uses for cards and modals, Bootstrap also provides a lot of tools for creating forms. It seems the client already started looking at form classes when they built the modal's form for creating a new task, but didn't they take it all the way.

Let's add the Bootstrap `form-control` class to both the `<textarea>` and `<input>` elements in the modal's form. Update each form field so that the code looks like this:

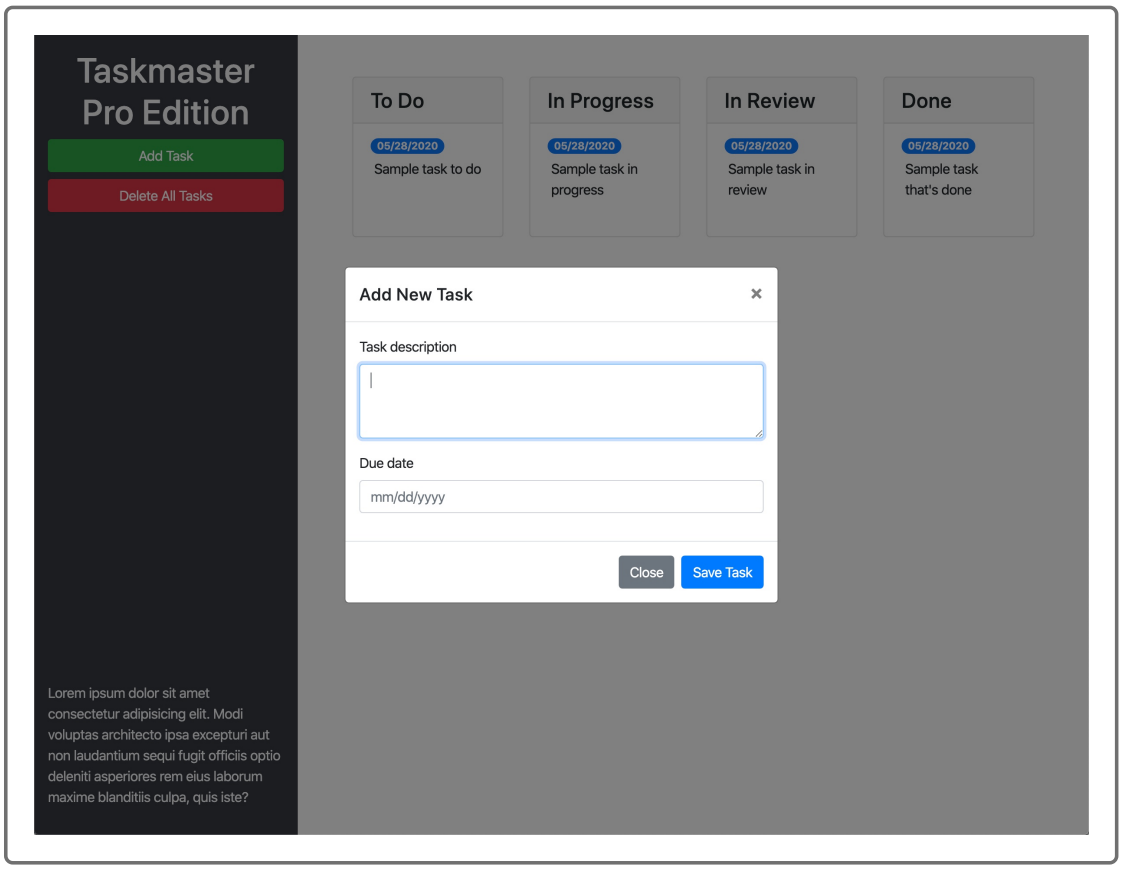
```
<div class="form-group">
  <label for="modalTaskDescription">Task description</label>
  <textarea id="modalTaskDescription" rows="3" class="form-control"></
</div>
<div class="form-group">
  <label for="modalDueDate">Due date</label>
  <input type="text" class="form-control" id="modalDueDate" placeholde
</div>
```

Then save the file, refresh the page, and open the modal.

HIDE HINT

The form we're updating is in the modal's HTML code towards the bottom of `index.html`.

The resulting form input elements should look a lot cleaner than they were before, as evident in the following image:



Congratulations on using both Bootstrap's grid system and UI components! Not too hard, right? Just study the documentation when you want to add something new and adjust the code until it fits your needs.

We need to address a few more things—mostly smaller details and refinements. Again, Bootstrap has an answer for that. Let's save the feature branch and push to GitHub so that we can move on!