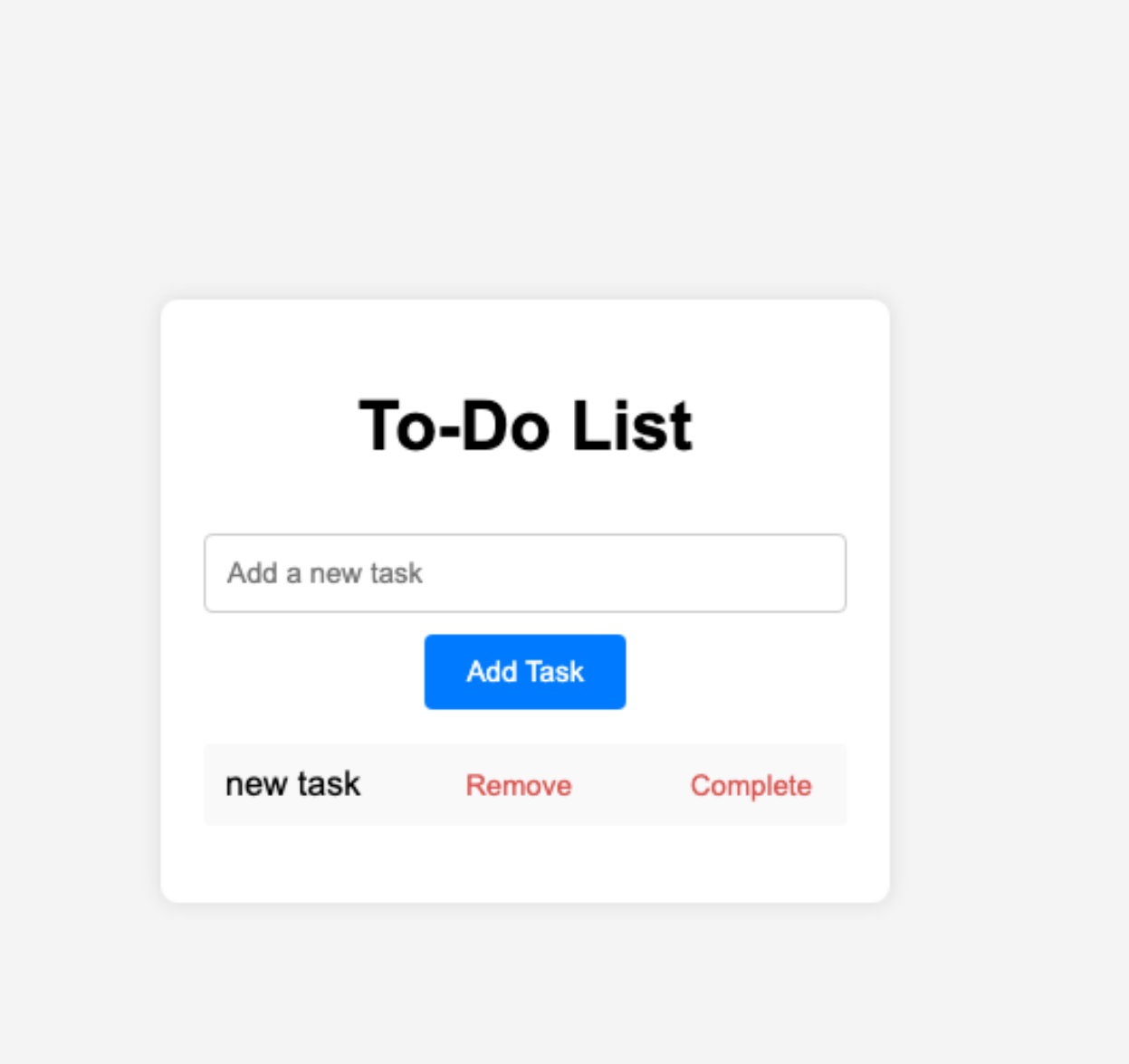
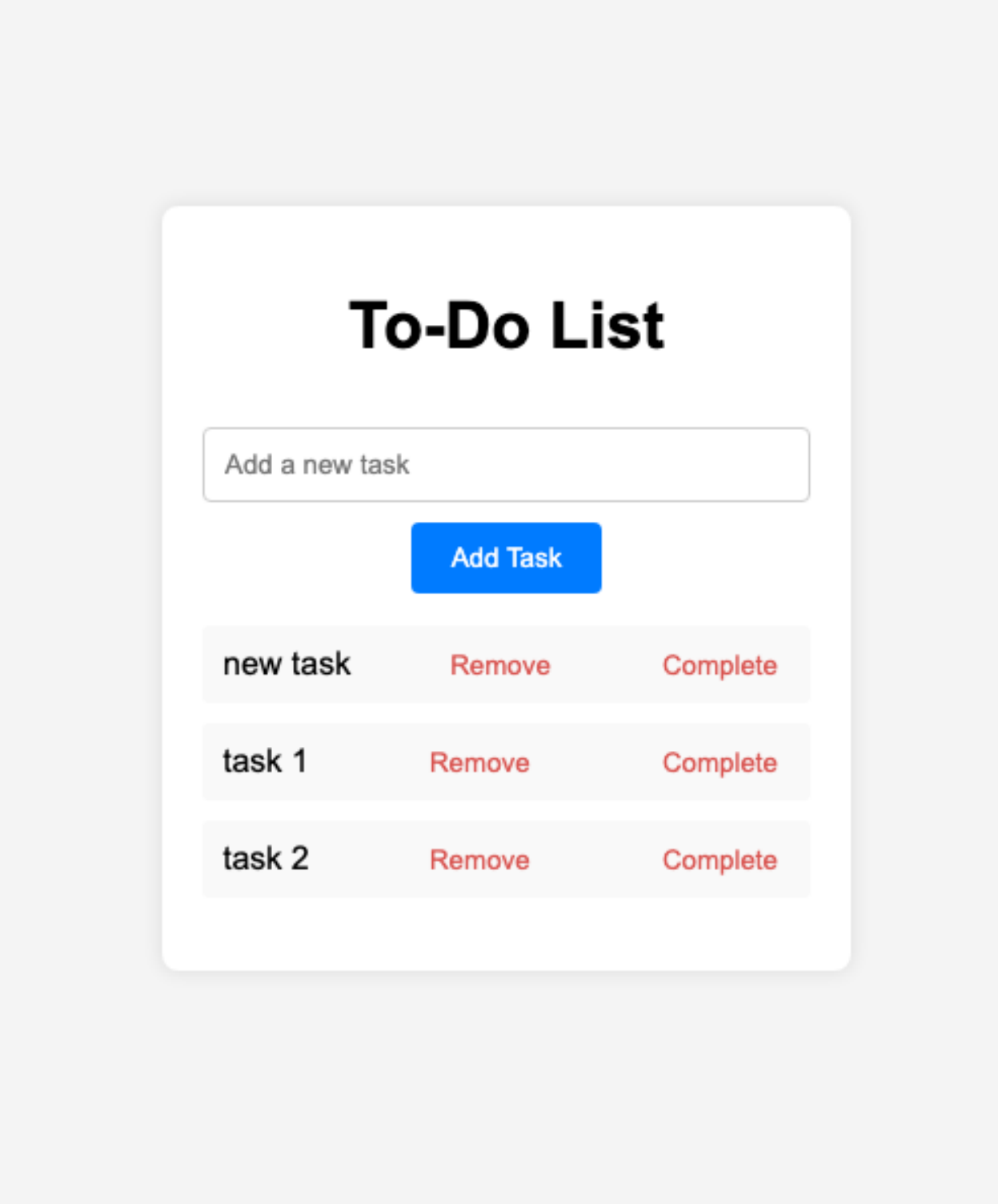
Week 15 Exercise.

Using JS to perform tasks like the following screen:





Answer the following questions: (Write “answer” below each question)

1. List CSS component to control position of the box

Answer position: This property allows you to specify the positioning method of an element. Values can be static (default), relative, absolute, fixed, or sticky. For example, if you set position: relative on a box and then use top, right, bottom, and left properties to move it relative to its normal position.

top, right, bottom, left: These properties are used in conjunction with the position property to precisely position an element. For instance, top: 20px will move the element 20 pixels down from its normal position when position is set to relative or absolute.

display: Can affect the layout and position of elements. For example, display: flex can be used to create a flexible box layout, allowing you to easily position child elements within the flex container.

float: Allows an element to be taken out of the normal flow and floated to the left or right of its container. Other elements will then flow around it.

margin and padding: These can be used to create space around the box and thus influence its position relative to other elements. For example, margin-top: 10px will add 10 pixels of space above the box.

2. List CSS component to add button
   Answer button selector: You can style the default browser button using the button CSS selector. For example, you can change the background color, text color, border, etc. like this: button { background-color: blue; color: white; border: none; }
   class and id: You can assign a class or an id to the button and then style it using the corresponding selector. For example, if you have a button with the class "myButton", you can style it like this: .myButton { font-size: 16px; padding: 10px 20px; }
   :hover, :active, :focus pseudo-classes: These allow you to change the style of the button when the user hovers over it, clicks it, or focuses on it. For example, button:hover { background-color: green; } will change the button's background color when the mouse hovers over it.
3. List CSS component to add “list of tasks”

Answer ul and li: The unordered list (ul) and list item (li) elements are used to create a basic list. You can style the list items as tasks. For example:

```
css
ul {
  list-style-type: none;
  padding: 0;
  margin: 0;
}

li {
  border: 1px solid #ccc;
  padding: 10px;
  margin-bottom: 5px;
}
```

display: flex or display: grid: If you want to create a more complex layout for the list of tasks, you can use these display properties to arrange the list items in a row or a grid. For example:

```
css
ul {
  display: flex;
  flex-wrap: wrap;
}

li {
  flex: 1 0 200px;
  margin: 5px;
}
```

checkbox and label: If your tasks have a checkbox to mark them as completed, you can style the checkbox and its associated label. For example:

```
css
input[type="checkbox"] {
  margin-right: 10px;
}
```

```
label {
  cursor: pointer;
}
```

4. List all JS variable and constants. Provide name and its purpose in your program
   Answer const tasks = [];: This is a constant that holds an array of tasks. The purpose is to store and manage the list of tasks in the application. It is declared as a constant because the reference to the array should not change throughout the program.
   let currentTaskIndex = 0;: This variable is used to keep track of the currently selected or active task in the list. It allows you to access and manipulate the specific task at a given time. The use of let allows the value to change as the user interacts with the tasks.
   const taskInput = document.getElementById('taskInput');: This constant stores a reference to the HTML input element where the user enters a new task. It is used to access the value entered by the user and perform operations like adding a new task to the tasks array.
5. List at least one control structure. Suggest one alternative structure to achieve the same purpose.
   Answer Control structure: A common control structure used is the for loop. For example, if you want to loop through the tasks array and display each task in the console, you can use a for loop like this:
   javascript
   ```
   for (let i = 0; i < tasks.length; i++) {
     console.log(tasks[i]);
   }
   ```
   Alternative structure: An alternative to the for loop in this case could be the forEach method. It provides a more concise way to iterate over an array. For example:
   javascript
   ```
   tasks.forEach(task => console.log(task));
   ```
6. List one JS event available in your program. Explain its purpose.
   6. Answer **JS event and its purpose**:
      - The click event is commonly used in programs. For example, when a user clicks a button to add a new task, you can listen for the click event on the button. The purpose is to trigger a specific action when the user interacts

with the element. In the case of the add task button, the click event can be used to get the value from the task input field and add it to the tasks array. Here is an example:

收起

```
javascript
const addTaskButton = document.getElementById('addTaskButton');
addTaskButton.addEventListener('click', () => {
  const newTask = taskInput.value;
  if (newTask) {
    tasks.push(newTask);
    taskInput.value = '';
    renderTasks();
  }
});
```

7. Explain the relevance of DOM model to your JS. List at least 3 relevancies.
   Answer Manipulating the structure: The DOM allows JavaScript to access and modify the HTML structure of a web page. For example, you can create new elements, add them to the page, remove existing elements, or change the attributes and content of elements. This is crucial for creating dynamic web applications where the content and layout can change based on user actions or data updates.
   Event handling: JavaScript can use the DOM to attach event handlers to HTML elements. As mentioned earlier, events like click, mouseover, keydown, etc. can be listened to and responded to. This enables interactivity in web pages, such as showing a dropdown menu when the user clicks a button or validating form input when the user types.
   Data binding: The DOM can be used to display data stored in JavaScript variables and update the UI when the data changes. For example, when a new task is added to the tasks array, JavaScript can update the DOM to display the new task in the list. This two-way data binding between the JavaScript data and the DOM is essential for creating responsive and interactive web applications.

Provide your html, CSS and JS.

Save this doc as pdf, and copy and paste to online text