

This screenshot shows the initial HTML structure of a To-Do list application in VS Code. The Explorer sidebar on the left lists the project files: shopping.css, shopping.html, todolist.css, todolist.html, todolist.js, and validation.js. The main editor displays the content of todolist.html, which includes a DOCTYPE declaration, a viewport meta tag, a title 'Simple To-Do List', and a link to todolist.css. The body contains a container div with a heading, an input field with a placeholder 'Enter task...', an 'Add Task' button, and an empty task list with id 'taskList'. A script tag at the bottom loads todolist.js.

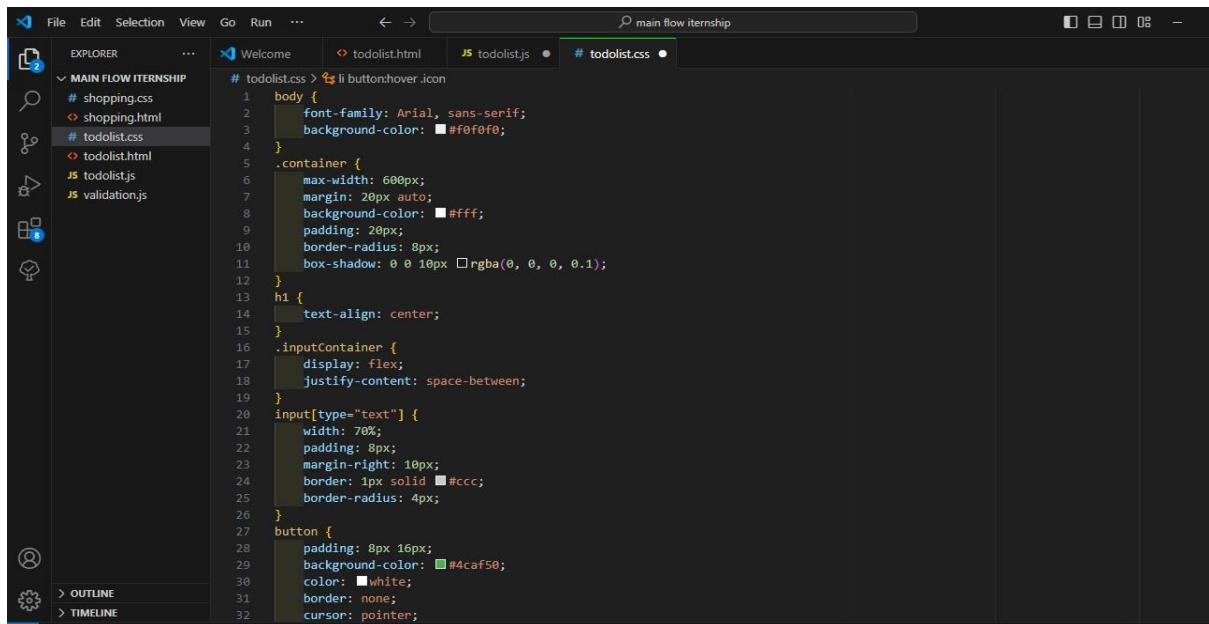
```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Simple To-Do List</title>
7   <link rel="stylesheet" href="todolist.css">
8 </head>
9 <body>
10  <div class="container">
11    <h1>Simple To-Do List</h1>
12    <div class="inputContainer">
13      <input type="text" id="taskInput" placeholder="Enter task...">
14      <button onclick="addTask()">Add Task</button>
15    </div>
16    <ul id="taskList"></ul>
17  </div>
18  <script src="todolist.js"></script>
19 </body>
20 </html>
21
```

This screenshot shows the JavaScript logic for adding tasks in todolist.js. The code starts by initializing an empty tasks array and adding an event listener for 'DOMContentLoaded' to call fetchTasks(). The fetchTasks() function retrieves the taskList element, clears its content, and iterates over the tasks array to add each task to the DOM. The addTask() function gets the task input, trims its value, and if it's not empty, pushes it to the tasks array and calls addTaskToDOM(). The addTaskToDOM() function creates a list item (li) with a class 'completed' if the task is completed, an input checkbox, and a span for the task text. It also creates a delete button and appends all these elements to the taskList.

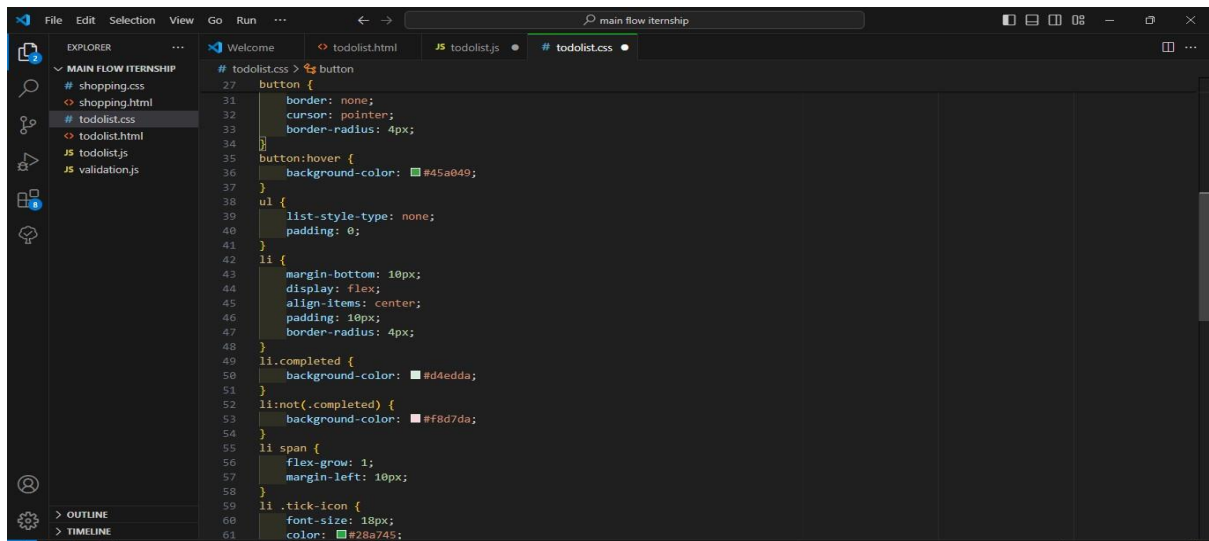
```
1 let tasks = [];
2 document.addEventListener('DOMContentLoaded', () => {
3   fetchTasks();
4 });
5 function fetchTasks() {
6   const taskList = document.getElementById('taskList');
7   taskList.innerHTML = '';
8   tasks.forEach((task, index) => {
9     addTaskToDOM(task, index);
10  });
11 }
12 function addTask() {
13   const taskInput = document.getElementById('taskInput');
14   const taskText = taskInput.value.trim();
15   if (taskText !== '') {
16     const task = { text: taskText, completed: false };
17     tasks.push(task);
18     addTaskToDOM(task, tasks.length - 1);
19     taskInput.value = '';
20   }
21 }
22 function addTaskToDOM(task, index) {
23   const taskList = document.getElementById('taskList');
24   const li = document.createElement('li');
25   li.className = task.completed ? 'completed' : '';
26   const checkbox = document.createElement('input');
27   checkbox.type = 'checkbox';
28   checkbox.checked = task.completed;
29   checkbox.onclick = () => toggleTaskCompletion(index);
30   const span = document.createElement('span');
31   span.textContent = task.text;
32   const deleteButton = document.createElement('button');
```

This screenshot shows the JavaScript logic for updating the DOM and deleting tasks in todolist.js. The updateDOM() function iterates over the tasks array and calls addTaskToDOM() for each task. The toggleTaskCompletion() function toggles the 'completed' status of a task and calls updateDOM(). The deleteTask() function removes a task from the tasks array and calls updateDOM(). The updateDOM() function retrieves the taskList element, clears its content, and iterates over the tasks array to add each task to the DOM.

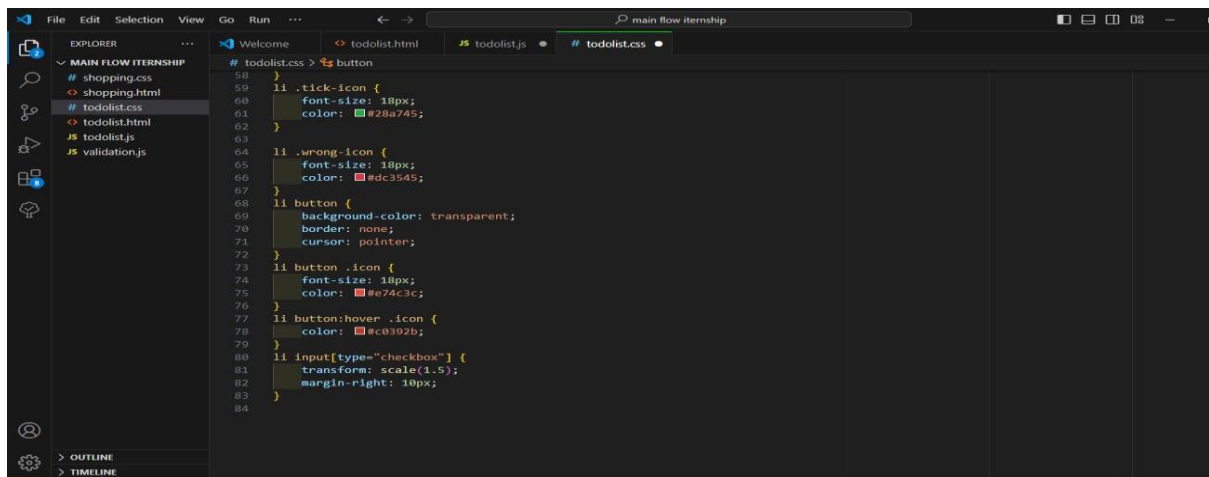
```
22 function addTaskToDOM(task, index) {
23   const deleteButton = document.createElement('button');
24   deleteButton.onclick = (e) => {
25     e.stopPropagation();
26     deleteTask(index);
27   };
28   deleteButton.innerHTML = '<span class="icon">X</span>';
29   li.appendChild(deleteButton);
30   taskList.appendChild(li);
31   if (task.completed) {
32     const tickIcon = document.createElement('span');
33     tickIcon.className = 'tick-icon icon';
34     tickIcon.innerHTML = '✓';
35     li.insertBefore(tickIcon, span);
36   }
37 }
38 function toggleTaskCompletion(index) {
39   tasks[index].completed = !tasks[index].completed;
40   updateDOM();
41 }
42 function deleteTask(index) {
43   tasks.splice(index, 1);
44   updateDOM();
45 }
46 function updateDOM() {
47   const taskList = document.getElementById('taskList');
48   taskList.innerHTML = '';
49   tasks.forEach((task, index) => {
50     addTaskToDOM(task, index);
51   });
52 }
```



```
# todolist.css > li button: hover .icon
1 body {
2   font-family: Arial, sans-serif;
3   background-color: #f0f0f0;
4 }
5 .container {
6   max-width: 600px;
7   margin: 20px auto;
8   background-color: #fff;
9   padding: 20px;
10  border-radius: 8px;
11  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
12 }
13 h1 {
14   text-align: center;
15 }
16 .inputContainer {
17   display: flex;
18   justify-content: space-between;
19 }
20 input[type="text"] {
21   width: 70%;
22   padding: 8px;
23   margin-right: 10px;
24   border: 1px solid #ccc;
25   border-radius: 4px;
26 }
27 button {
28   padding: 8px 16px;
29   background-color: #4caf50;
30   color: white;
31   border: none;
32   cursor: pointer;
```

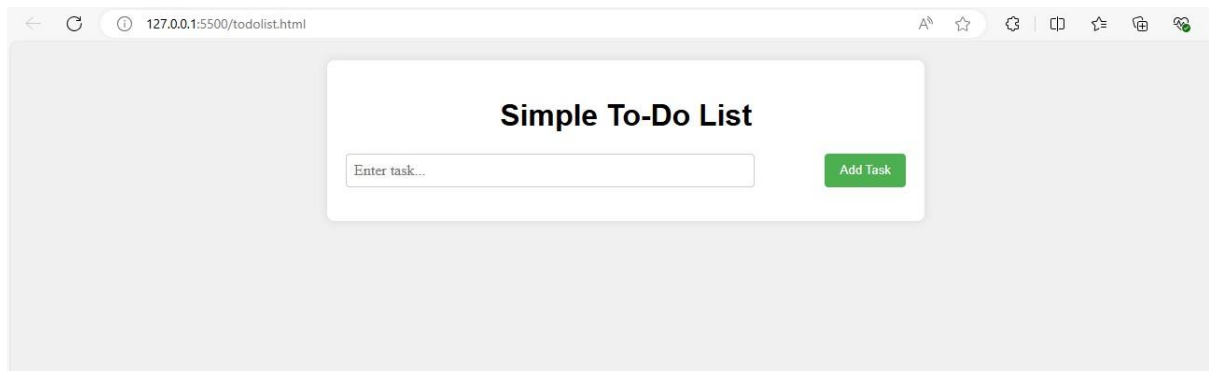


```
# todolist.css > button
27 button {
31   border: none;
32   cursor: pointer;
33   border-radius: 4px;
34 }
35 button: hover {
36   background-color: #45a049;
37 }
38 ul {
39   list-style-type: none;
40   padding: 0;
41 }
42 li {
43   margin-bottom: 10px;
44   display: flex;
45   align-items: center;
46   padding: 10px;
47   border-radius: 4px;
48 }
49 li.completed {
50   background-color: #d4edda;
51 }
52 li:not(.completed) {
53   background-color: #f8d7da;
54 }
55 li span {
56   flex-grow: 1;
57   margin-left: 10px;
58 }
59 li .tick-icon {
60   font-size: 18px;
61   color: #28a745;
```

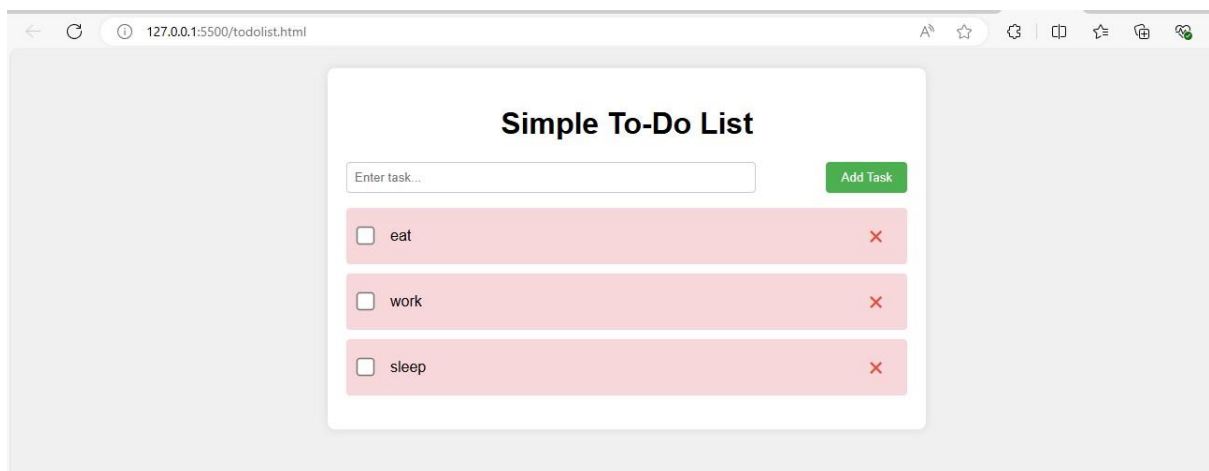
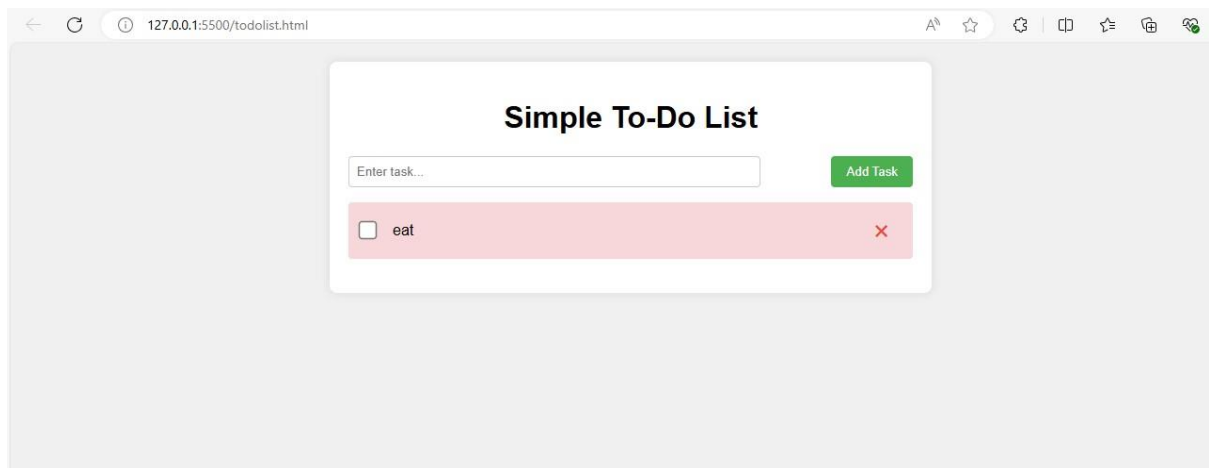


```
# todolist.css > button
59 }
60 li .tick-icon {
61   font-size: 18px;
62   color: #28a745;
63 }
64 li .wrong-icon {
65   font-size: 18px;
66   color: #dc3545;
67 }
68 li button {
69   background-color: transparent;
70   border: none;
71   cursor: pointer;
72 }
73 li button .icon {
74   font-size: 18px;
75   color: #e74c3c;
76 }
77 li button: hover .icon {
78   color: #c0392b;
79 }
80 li input[type="checkbox"] {
81   transform: scale(1.5);
82   margin-right: 10px;
83 }
84 }
```

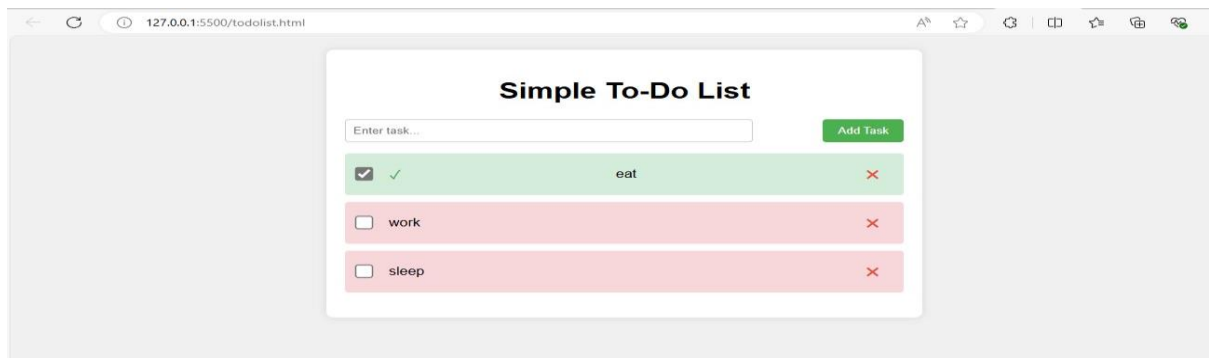
To-do list



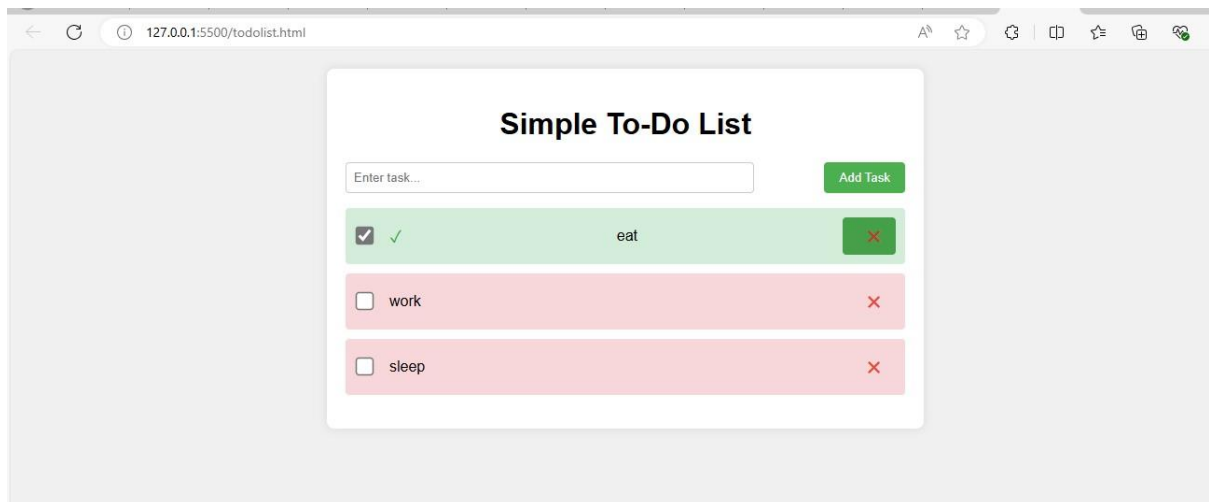
After entering the tasks



If the task is completed



Deleting the task



After deleting the task

