

# **Project Title**

## **To-Do List App with Local Storage**

### **Objective**

- To design and develop a dynamic To-Do
- List web application that allows users to:
- Create new tasks
- Update existing tasks
- Delete tasks
- Store tasks using LocalStorage so data remains persistent even after page refresh

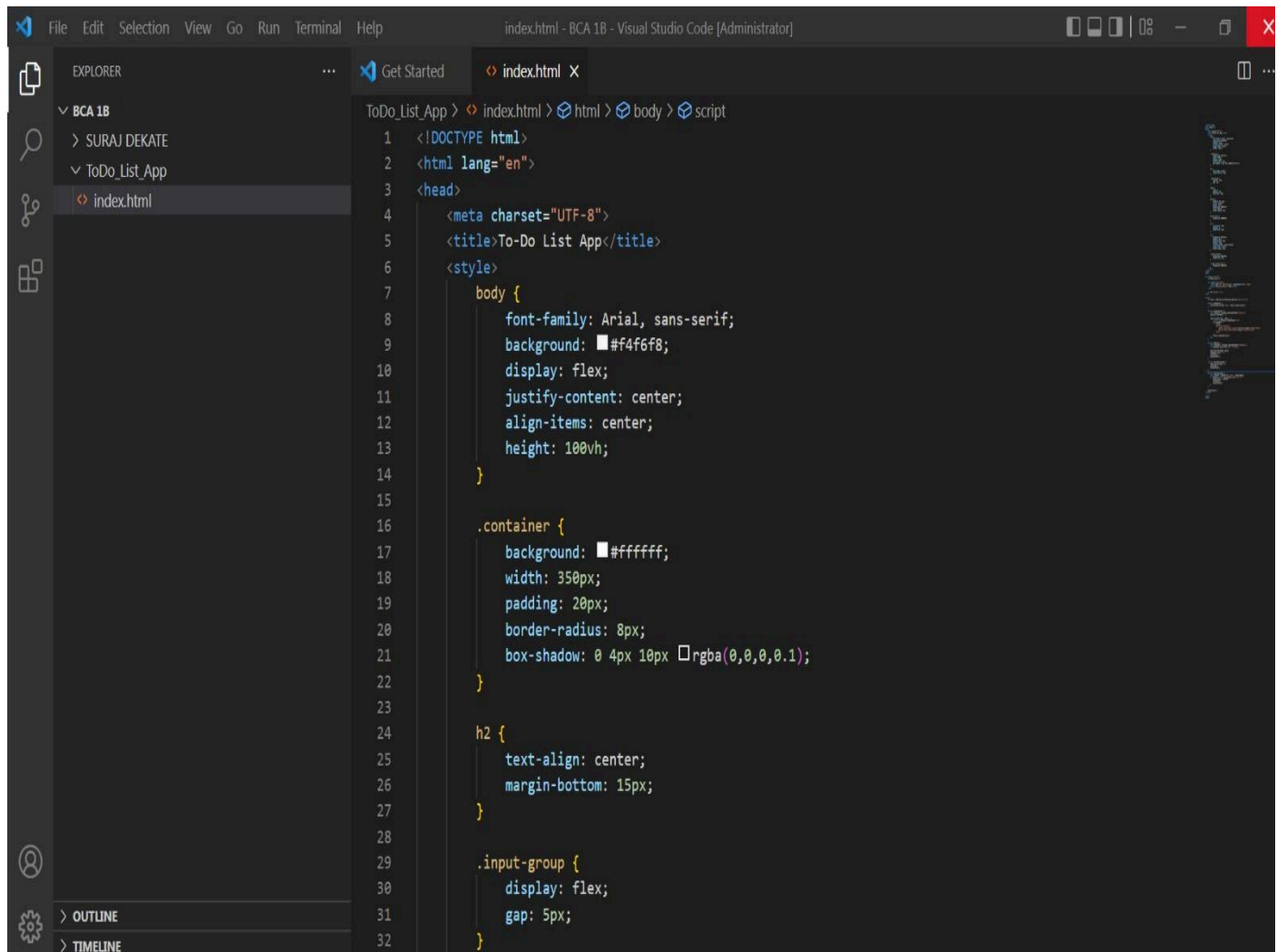
### **Tech Stack**

- HTML – Structure
- CSS – Styling (clean & professional UI)
- JavaScript – Logic & interactivity
- LocalStorage – Persistent data storage

# Expected Output:

- User can add, edit, and delete tasks
- Tasks remain saved even after browser refresh

# Program:



```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>To-Do List App</title>
6   <style>
7     body {
8       font-family: Arial, sans-serif;
9       background: #f4f6f8;
10      display: flex;
11      justify-content: center;
12      align-items: center;
13      height: 100vh;
14    }
15
16    .container {
17      background: #ffffff;
18      width: 350px;
19      padding: 20px;
20      border-radius: 8px;
21      box-shadow: 0 4px 10px rgba(0,0,0,0.1);
22    }
23
24    h2 {
25      text-align: center;
26      margin-bottom: 15px;
27    }
28
29    .input-group {
30      display: flex;
31      gap: 5px;
32    }
```

This screenshot shows the CSS styles for the To-Do List App in VS Code. The Explorer panel on the left shows the project structure: BCA 18 > SURAJ DEKATE > ToDo\_List\_App > index.html. The breadcrumb navigation at the top of the editor shows the path: index.html > html > body > script. The main editor area displays the following CSS code:

```
30 display: flex;
31 gap: 5px;
32 }
33
34 input {
35   flex: 1;
36   padding: 8px;
37   font-size: 14px;
38 }
39
40 button {
41   padding: 8px 12px;
42   cursor: pointer;
43   border: none;
44   background: #007bff;
45   color: white;
46   border-radius: 4px;
47 }
48
49 button:hover {
50   background: #0056b3;
51 }
52
53 ul {
54   list-style: none;
55   padding: 0;
56   margin-top: 15px;
57 }
58
59 li {
60   background: #f1f1f1;
61   padding: 8px;
```

This screenshot shows the HTML structure for the To-Do List App in VS Code. The Explorer panel on the left shows the project structure: BCA 18 > SURAJ DEKATE > ToDo\_List\_App > index.html. The breadcrumb navigation at the top of the editor shows the path: index.html > html > body > script. The main editor area displays the following HTML code:

```
59 li {
60   background: #f1f1f1;
61   padding: 8px;
62   margin-bottom: 8px;
63   display: flex;
64   justify-content: space-between;
65   align-items: center;
66   border-radius: 4px;
67 }
68
69 .actions button {
70   background: #dc3545;
71   margin-left: 5px;
72 }
73
74 .actions button.edit {
75   background: #28a745;
76 }
77 </style>
78 </head>
79 <body>
80
81 <div class="container">
82   <h2>To-Do List</h2>
83
84   <div class="input-group">
85     <input type="text" id="taskInput" placeholder="Enter a task">
86     <button onclick="addTask()">Add</button>
87   </div>
88
89   <ul id="taskList"></ul>
90 </div>
```

```
91 <script>
92   let tasks = JSON.parse(localStorage.getItem("tasks")) || [];
93
94   function saveTasks() {
95     localStorage.setItem("tasks", JSON.stringify(tasks));
96   }
97
98   function displayTasks() {
99     const taskList = document.getElementById("taskList");
100     taskList.innerHTML = "";
101
102     tasks.forEach((task, index) => {
103       const li = document.createElement("li");
104       li.innerHTML = `
105         ${task}
106         <div class="actions">
107           <button class="edit" onclick="editTask(${index})">Edit</button>
108           <button onclick="deleteTask(${index})">Delete</button>
109         </div>
110       `;
111       taskList.appendChild(li);
112     });
113   }
114
115   function addTask() {
116     const taskInput = document.getElementById("taskInput");
117     if (taskInput.value.trim() === "") return;
118
119     tasks.push(taskInput.value);
120     taskInput.value = "";
121     saveTasks();
122   }
```

```
120     tasks.push(taskInput.value);
121     taskInput.value = "";
122     saveTasks();
123     displayTasks();
124   }
125
126   function deleteTask(index) {
127     tasks.splice(index, 1);
128     saveTasks();
129     displayTasks();
130   }
131
132   function editTask(index) {
133     const newTask = prompt("Edit task:", tasks[index]);
134     if (newTask !== null && newTask.trim() !== "") {
135       tasks[index] = newTask;
136       saveTasks();
137       displayTasks();
138     }
139   }
140
141   displayTasks();
142 </script>
143
144 </body>
145 </html>
```

# Output:

