

Experiment 5: Perform DOM Manipulation by Developing a Dynamic To-Do List Application Using JavaScript

HTML CODE:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Dynamic To-Do List</title>
  <link rel="stylesheet" href="style5.css">
</head>
<body>

  <div class="container">
    <h1>My To-Do List</h1>

    <div class="input-section">
      <input type="text" id="taskInput"
placeholder="Enter a new task">
      <button id="addBtn">Add
Task</button>
    </div>

    <ul id="taskList"></ul>
  </div>

  <script src="script5.js"></script>
</body>
</html>
```

CSS CODE:

```
body {
  font-family: Verdana, sans-serif;
  background: linear-gradient(to right,
#1e3c72, #2a5298);
}

.container {
  width: 420px;
  margin: 70px auto;
  text-align: center;
  background: #ffffff;
```

```
padding: 25px;
border-radius: 10px;
box-shadow: 0 0 15px #00000050;
}

h1 {
  color: #2a5298;
}

input {
  width: 70%;
  padding: 8px;
  border: 2px solid #2a5298;
  border-radius: 4px;
}

button {
  padding: 8px 12px;
  cursor: pointer;
  background-color: #2a5298;
  color: white;
  border: none;
  border-radius: 4px;
}

button:hover {
  background-color: #1e3c72;
}

ul {
  list-style-type: none;
  padding: 0;
  margin-top: 20px;
}

li {
  background: #e3f2fd;
  margin: 8px 0;
  padding: 10px;
```

```

display: flex;
justify-content: space-between;
align-items: center;
border-radius: 5px;
}

```

```

.deleteBtn {
background: #d32f2f;
color: white;
border: none;
padding: 5px 8px;
border-radius: 3px;
cursor: pointer;
}

```

```

.deleteBtn:hover {
background: #b71c1c;
}

```

JAVASCRIPT CODE:

```

const taskInput =
document.getElementById("taskInput");
const addBtn =
document.getElementById("addBtn");
const taskList =
document.getElementById("taskList");

```

```

function addTask() {
const taskText = taskInput.value.trim();

if (taskText === "") {
alert("Please enter a task!");
return;
}

```

```

const li = document.createElement("li");
li.innerText = taskText;

```

```

const deleteBtn =
document.createElement("button");
deleteBtn.innerText = "Delete";
deleteBtn.classList.add("deleteBtn");

```

```

deleteBtn.addEventListener("click",
function () {
taskList.removeChild(li);
});

```

```

li.appendChild(deleteBtn);
taskList.appendChild(li);

```

```

taskInput.value = "";
}

```

```

addBtn.addEventListener("click", addTask);

```

```

taskInput.addEventListener("keydown",
function (event) {
if (event.key === "Enter") {
addTask();
}
});

```

OUTPUT:

The screenshot shows a web application titled "My To-Do List". It has a dark blue border. Inside, there's a white background. At the top, the title "My To-Do List" is displayed in a large, bold, dark blue font. Below the title, there's a text input field with a light blue border and a placeholder text "Enter a new task". To the right of the input field is a blue button with white text that says "Add Task". Below these elements is a light blue rectangular box. Inside this box, the text "complete all the post experiment answers" is written in a dark blue font. To the right of this text is a red button with white text that says "Delete".