Program - BSc (Hons) Computer Systems Engineering (BSc.IT)

NAME: Aarjan Paudel

STUDENT NUMBER: 240703679

MODULE CODE: CET138

MODULE TITLE: Full Stack Development

MODULE LEADER: David Grey

SUBMISSION DATE AND TIME: As stated in the assignment submission page on Canvas for this module

ASSIGNMENT: First Assignment

Academic Misconduct is an offence under university regulations, and this involves:

- **Plagiarism** – where you use information from another information source (including your previously submitted work) and pass it off as your own. This can be through direct copying, poor paraphrasing and/or absence of citations.
- **Collusion** – where you work too closely, intentionally, or unintentionally, with others to produce work that is similar in nature. This can be through loaning of materials, drafts or through unauthorised use of a fellow student's work.
- **Asking another person to write your assignment** – where you ask another individual or company to complete your work for you, be that paid or unpaid, and submit it as if it were your own.
- **Unauthorised use of artificial intelligence** – where you use artificial intelligence tools to generate your assignment instead of completing it yourself and/or where you have not been given permission to use artificial intelligence tools by your module leader. Please complete the following declaration around you use of artificial intelligence tools in your assignment.

STATEMENT ON USE OF ARTIFICIAL INTELLIGENCE TOOLS:

• I have used artificial intelligence tools to generate an idea for my assignment:	**NO**
• I have used artificial intelligence tools to write my assignment for me:	**NO**
• I have used artificial intelligence tools to brainstorm ideas for my assignment:	**NO**
• I have used artificial intelligence tools to correct my original assignment:	**NO**

DECLARATION

- I understand that by submitting this piece of work I am declaring it to be my own work and in compliance with the university regulations on Academic Integrity.
- I confirm that I have done this work myself without external support or inappropriate use of resources.
- I understand that I am only permitted to use artificial intelligence tools in line with guidance provided by my Module Leader, and I have not used artificial intelligence tools outside this remit.
- I confirm that this piece of work has not been submitted for any other assignment at this or another institution prior to this point in time.
- I can confirm that all sources of information, including quotations, have been acknowledged by citing the source in the text, along with producing a full list of the sources used at the end of the assignment.
- I understand that academic misconduct is an offence and can result in formal disciplinary proceedings.
- I understand that by submitting this assignment, I declare myself fit to be able to complete the assignment and I accept the outcome of the assessment as valid and appropriate.

ANY OTHER COMMENTS FROM STUDENT:

JAVASCRIPT

What is JavaScript?

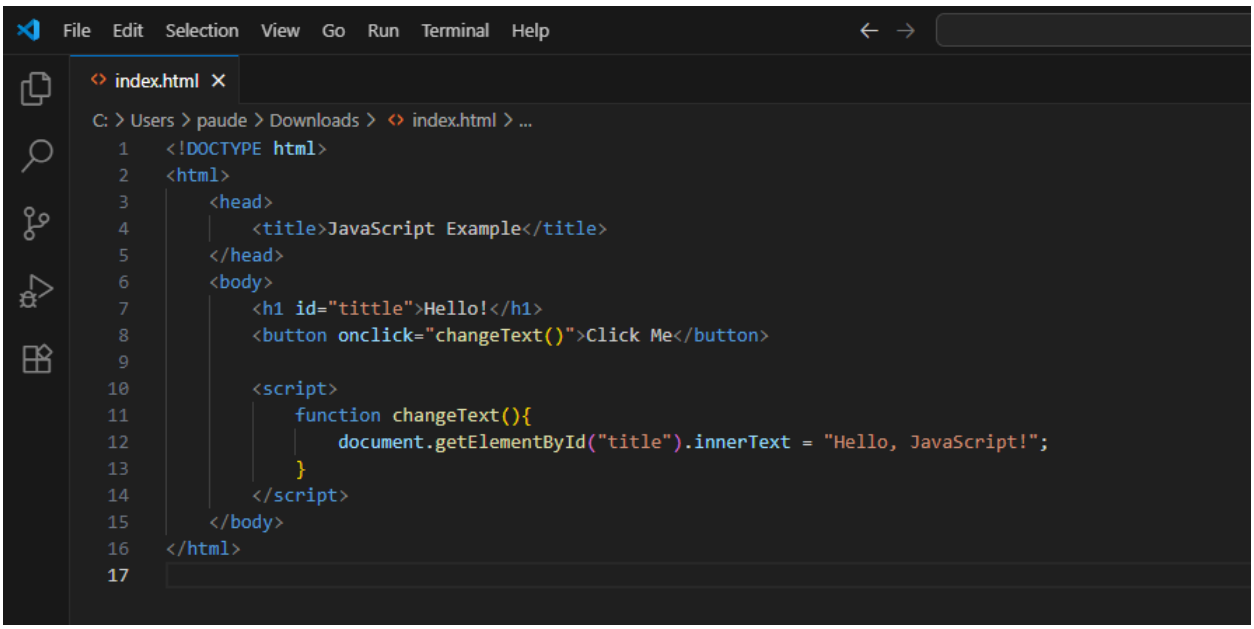
JavaScript(JS) is a high-level programming language used to add interactivity and dynamic behavior to a website.

- While **HTML** creates a web page's structure and **CSS** creates the design, **JavaScript** makes web pages interactive (e.g., dynamic results, animations, form validation, calculators, games, to-do lists).
- It's all being executed in the browser, making it interactive in real-time without needing to refresh.

Key Features of JavaScript

- **Dynamic Content** → Change website content on the fly (like show/hide sections).
- **Event Handling** → Reacts to user interactions such as clicks, keystrokes and whatnot.
- **DOM Manipulation** → It changes your HTML elements, styles, or attributes dynamically.
- **Built in Functions** → For math operations, manipulating strings, handling date/time.
- **Cross-Platform** → It works in all modern browsers.

Example: Simple JavaScript



How it Works

- When the button is clicked changeText() function is called.
- JavaScript uses document.getElementById() to find the <h1> element.
- The text content is changed from **Hello!** to **Hello, JavaScript!** dynamically.

My Task: JavaScript Projects

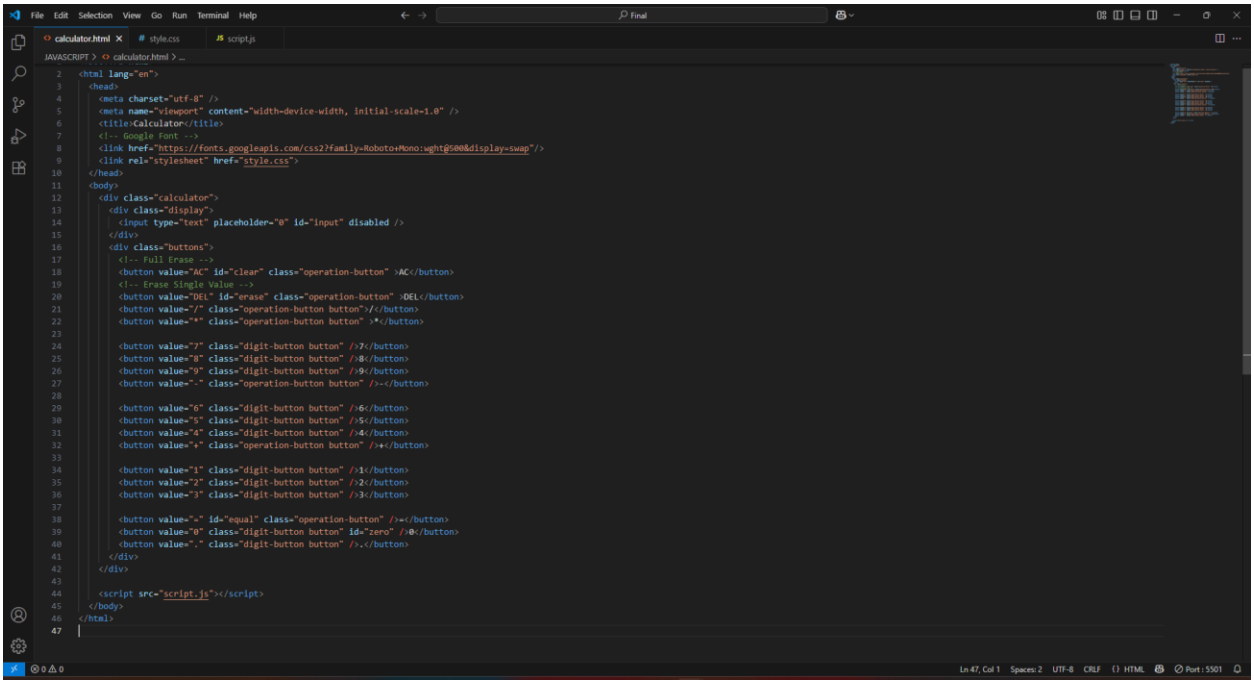
1. **Calculator**

Description:

- A Working calculator is built with use of HTML, CSS, JavaScript.
- It supports addition, subtraction, multiplication, division, decimal values, clear, and delete functions.

Calculator code:

a. **HTML code:**



```html
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Calculator</title>
    <!-- Google Font -->
    <link href="https://fonts.googleapis.com/css2?family=Roboto+Mono:wght@500&display=swap"/>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <div class="calculator">
      <div class="display">
        <input type="text" placeholder="0" id="input" disabled />
      </div>
      <div class="buttons">
        <!-- Full Erase -->
        <button value="AC" id="clear" class="operation-button" >AC</button>
        <!-- Erase Single Value -->
        <button value="DEL" id="erase" class="operation-button" >DEL</button>
        <button value="/" class="operation-button button">/</button>
        <button value="*" class="operation-button button" >*</button>

        <button value="7" class="digit-button button" />7</button>
        <button value="8" class="digit-button button" />8</button>
        <button value="9" class="digit-button button" />9</button>
        <button value="-" class="operation-button button" />-</button>

        <button value="6" class="digit-button button" />6</button>
        <button value="5" class="digit-button button" />5</button>
        <button value="4" class="digit-button button" />4</button>
        <button value="+" class="operation-button button" />+</button>

        <button value="1" class="digit-button button" />1</button>
        <button value="2" class="digit-button button" />2</button>
        <button value="3" class="digit-button button" />3</button>

        <button value="=" id="equal" class="operation-button" />=</button>
        <button value="0" class="digit-button button" id="zero" />0</button>
        <button value="." class="digit-button button" />.</button>
      </div>
    </div>

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

b. CSS code:

File Edit Selection View Go Run Terminal Help

calculator.html | style.css | script.js

JAVASCRIPT > style.css > a:hover

```
* {
  padding: 0;
  margin: 0;
  box-sizing: border-box;
  font-family: "Roboto Mono", monospace;
}
body {
  height: 100vh;
  background-color: #dadff7;
}
.calculator {
  width: 400px;
  background-color: #071013;
  padding: 50px 30px 20px 30px;
  position: absolute;
  transform: translate(-50%, -50%);
  top: 50%;
  left: 50%;
  border-radius: 8px;
  box-shadow: 0 20px 50px rgba(0, 5, 24, 0.4);
}
.display {
  width: 100%;
}
.display input {
  width: 100%;
  padding: 15px 10px;
  text-align: right;
  border: none;
  background-color: transparent;
  color: #ffffff;
  font-size: 35px;
}
.display input::placeholder {
  color: #9490ac;
}
.buttons {
  display: grid;
  grid-template-columns: repeat(4, 1fr);
  grid-gap: 20px;
  margin-top: 40px;
}
```

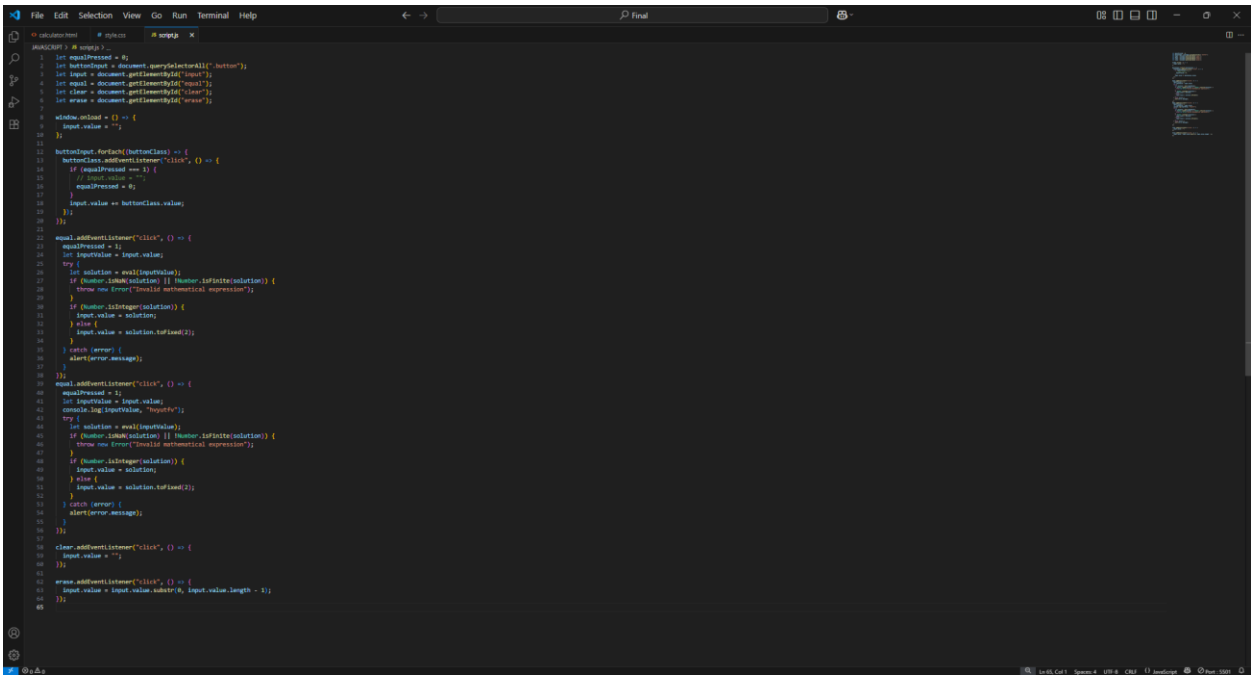
File Edit Selection View Go Run Terminal Help

calculator.html | # style.css | JS script.js

JAVASCRIPT > # style.css > a:hover

```
43  button {
44    font-size: 20px;
45    padding: 17px;
46    border: none;
47    color: #ffffff;
48    cursor: pointer;
49    border-radius: 5px;
50    border-radius: 10px;
51  }
52  button#equal {
53    grid-row: span 2;
54  }
55  button#zero {
56    grid-column: span 2;
57  }
58  .operation-button {
59    background-color: #f69906;
60  }
61  .operation-button:hover {
62    background-color: #eef606db;
63  }
64  .digit-button {
65    background-color: #313131;
66  }
67  .digit-button:hover {
68    background-color: #313131a8;
69  }
70  .main {
71    margin-top: 10%;
72  }
73  .brand {
74    text-align: center;
75    color: white;
76  }
77  a {
78    color: white;
79    font-weight: 800;
80  }
81  a:hover {
82    color: #f69906;
83  }
84
```

c. **JavaScript code:**



```javascript
let equalPressed = 0;
let buttonInput = document.querySelectorAll(".button");
let input = document.getElementById("input");
let equal = document.getElementById("equal");
let clear = document.getElementById("clear");
let erase = document.getElementById("erase");

window.onload = () => {
  input.value = "";
};

buttonInput.forEach(buttonClass) => {
  buttonClass.addEventListener("click", () => {
    if (equalPressed === 1) {
      // input.value = "";
      equalPressed = 0;
    }
    input.value += buttonClass.value;
  });
});

equal.addEventListener("click", () => {
  equalPressed = 1;
  let inputValue = input.value;
  try {
    let solution = eval(inputValue);
    if (Number.isNaN(solution) || !Number.isFinite(solution)) {
      throw new Error("Invalid mathematical expression");
    }
    if (Number.isInteger(solution)) {
      input.value = solution;
    } else {
      input.value = solution.toFixed(2);
    }
  } catch (error) {
    alert(error.message);
  }
});
equal.addEventListener("click", () => {
  equalPressed = 1;
  let inputValue = input.value;
  console.log(inputValue, "hsyutfv");
  try {
    let solution = eval(inputValue);
    if (Number.isNaN(solution) || !Number.isFinite(solution)) {
      throw new Error("Invalid mathematical expression");
    }
    if (Number.isInteger(solution)) {
      input.value = solution;
    } else {
      input.value = solution.toFixed(2);
    }
  } catch (error) {
    alert(error.message);
  }
});

clear.addEventListener("click", () => {
  input.value = "";
});

erase.addEventListener("click", () => {
  input.value = input.value.substr(0, input.value.length - 1);
});
```

How it works:

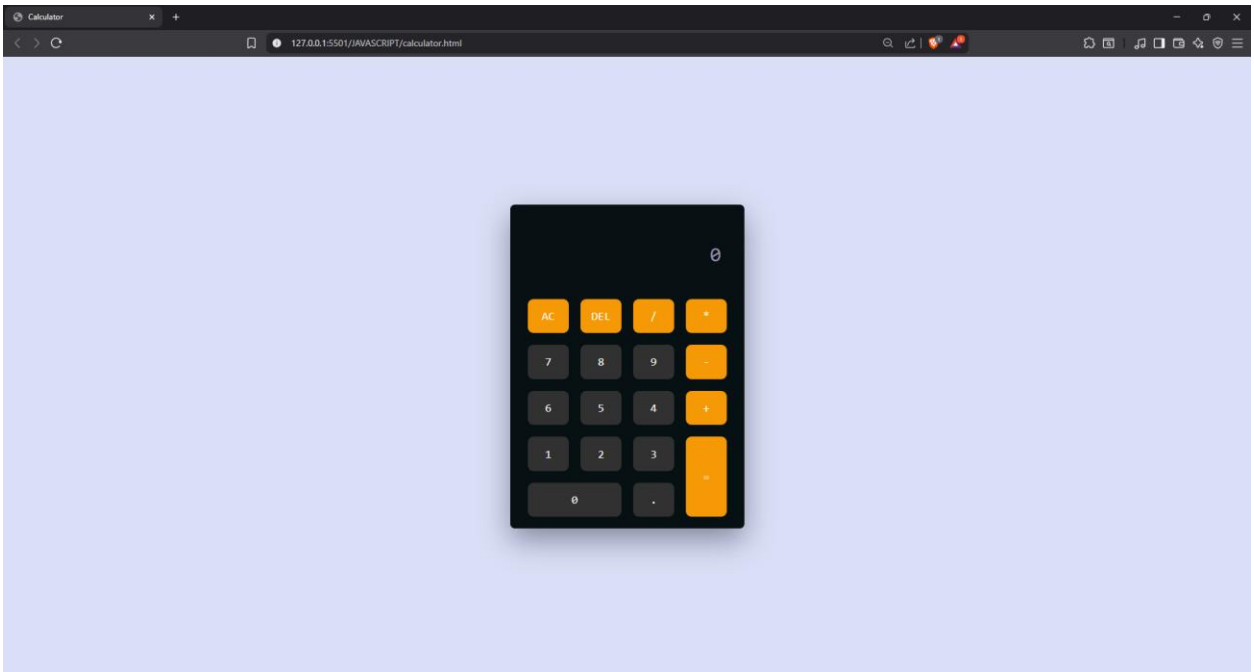
HTML: provides a solid structure with places for buttons and display screens

CSS: meanwhile has buttons arrangement layouts that inherit color properties from other buttons so they can be cut down on later adjustments

JavaScript should handle logic:

- Captures the button clicks.
- Parsing the entire transaction using eval().
- Displays result on input screen.
- The AC key clears all input.
- DEL removes that item from the last section if there is one present.
- Handling errors are also possible (e.g., invalid expressions that don't match anything expected).

Result:



What it demonstrates:

- Ability to handle events (button clicks).
- DOM manipulation in use (document.getElementById).
- Mathematical operations implementations are done dynamically.

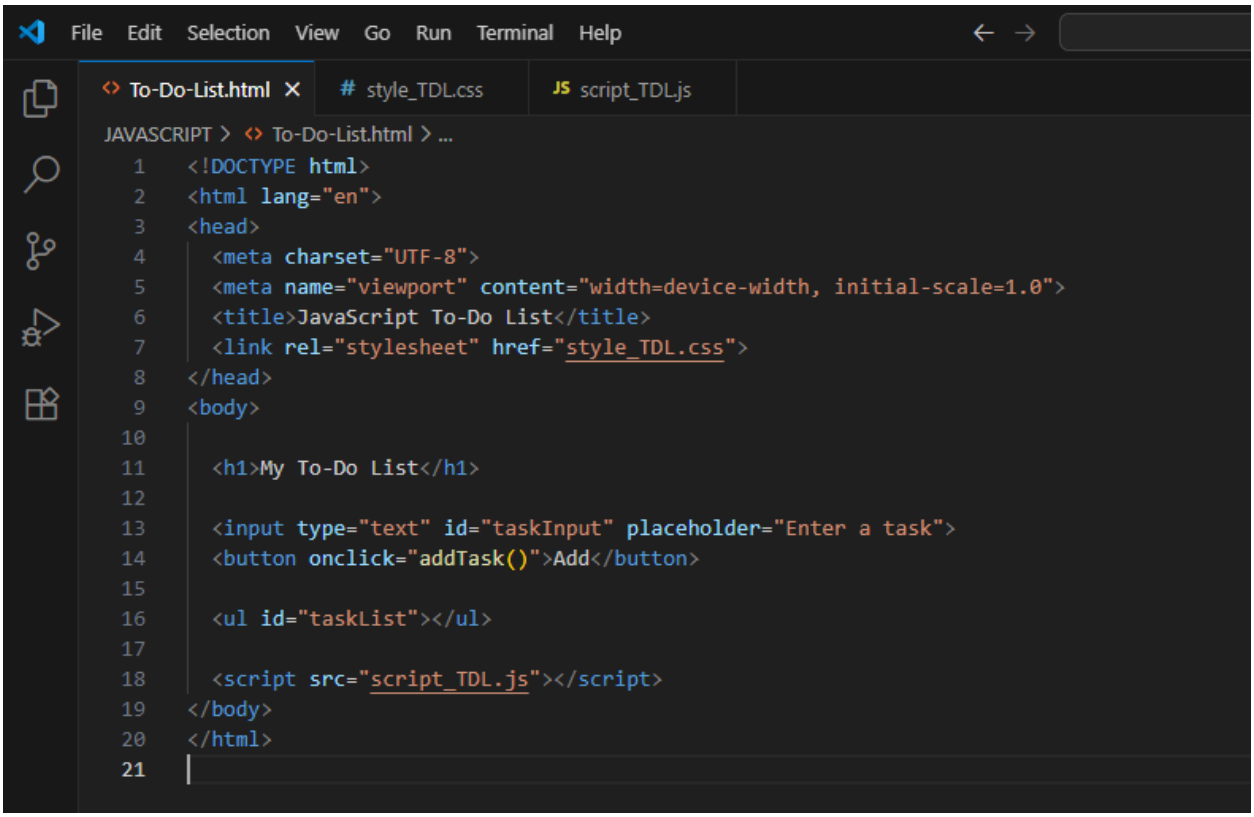
2. To-Do List

Description:

To-Do List with Add, Complete, and Delete Mechanics.

To-Do-List Code:

a. HTML code:




```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>JavaScript To-Do List</title>
  <link rel="stylesheet" href="style_TDL.css">
</head>
<body>

  <h1>My To-Do List</h1>

  <input type="text" id="taskInput" placeholder="Enter a task">
  <button onclick="addTask()">Add</button>

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

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

b. CSS code:

File Edit Selection View Go Run Terminal Help

To-Do-List.html | style_TDL.css | script_TDL.js

JAVASCRIPT > style_TDL.css > .delete-btn

```
body {
  font-family: Arial, sans-serif;
  text-align: center;
  background: #f4f4f4;
  padding: 40px;
}
input {
  padding: 10px;
  width: 200px;
}
button {
  padding: 10px;
  margin-left: 5px;
}
ul {
  list-style-type: none;
  padding: 0;
  margin-top: 20px;
}
li {
  margin: 8px auto;
  padding: 10px;
  background: #fff;
  width: 300px;
  display: flex;
  justify-content: space-between;
  align-items: center;
  border-radius: 5px;
  box-shadow: 2px 2px 6px rgba(0,0,0,0.1);
}
.done {
  text-decoration: line-through;
  color: grey;
}
.delete-btn {
  background: #e74c3c;
  color: white;
  border: none;
  padding: 5px 10px;
  border-radius: 3px;
  cursor: pointer;
}
```

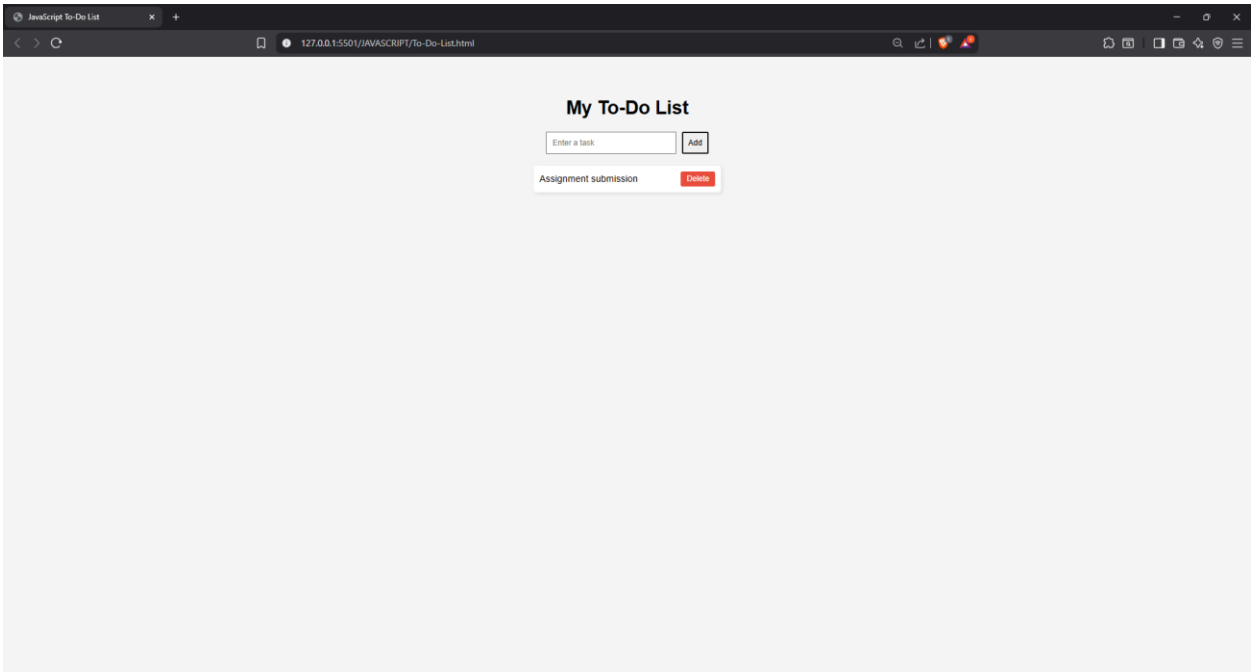
c. sJavaScript Code:

```javascript
function addTask() {
    const input = document.getElementById("taskInput");
    const task = input.value.trim();

    if (task === "") return; // ignore empty tasks

    // Create new list item
    const li = document.createElement("li");

    // Create span for task text
    const span = document.createElement("span");
    span.textContent = task;
    span.onclick = function() {
      span.classList.toggle("done"); // toggle completed
    };

    // Create delete button
    const deleteBtn = document.createElement("button");
    deleteBtn.textContent = "Delete";
    deleteBtn.className = "delete-btn";
    deleteBtn.onclick = function() {
      li.remove(); // remove task
    };

    // Append everything
    li.appendChild(span);
    li.appendChild(deleteBtn);
    document.getElementById("taskList").appendChild(li);

    // Clear input
    input.value = "";
  }
```

How it Works:

- **HTML** → Input field, Add button, and list of items(ul).
- **CSS** → Styles the task list with some shadows, spacing and a “done” effect via text-decoration: line-through.
- **JavaScript** → Adds interactivity:
    - addTask() makes a new <li> with the task test.
    - It toggles the “done” class (Completed) on clicking on the task.
    - The task is removed via delete button with .remove().
    - Prevents adding empty tasks with the following if (task === "") return;.

Result:



What it demonstrates:

- DOM Manipulation (Some dynamic element creation).
- Handling (Click to add or remove items).
- Conditionals and Functions in JavaScript.