INTERNSHIP REPORT

WEEK 5 DAY 1

Submitted to:	**Ali Hyder**	Submission Date:	**21ST July, 2025**
Internship Domain:	**Front Development**	Internship Name:	**ProSensia**
Student Name:	**Yasal Qamar**	Roll No.	**S25031**

JavaScript Advanced

Topics: Events (onclick, onchange, addEventListener)

Objective

To understand JavaScript event handling mechanisms, particularly how user interactions (clicks, changes, etc.) can trigger code using event handlers like onclick, onchange, and the more flexible addEventListener().

Topics Covered

1. onclick Event

- Triggered when a user clicks on an element (like a button).
- Inline or script-based event binding.

Example:

```
<button onclick="alert('Button clicked!')">Click Me</button>
```

2. onchange Event

- Fired when the value of an input, <select>, or <textarea> changes and loses focus.
- Commonly used with form elements.

Example:

```
<select onchange="alert(this.value)">
```

```
  <option value="HTML">HTML</option>

  <option value="CSS">CSS</option>

  <option value="JavaScript">JavaScript</option>

</select>
```

✅ 3. addEventListener()

- A modern, powerful method to attach multiple event handlers to a single element without overwriting previous ones.
- Syntax: element.addEventListener('event', function)

Example:

```
<button id="myBtn">Click</button>

<script>

  document.getElementById("myBtn").addEventListener("click", function () {

    alert("Handled by addEventListener!");

  });

</script>
```

Learning Outcome

- Gained hands-on experience with core JavaScript events used in real-world web applications.
- Understood the difference between inline (onclick, onchange) and modern (addEventListener) event handling.
- Learned to build a user-friendly interface that reacts dynamically to input.

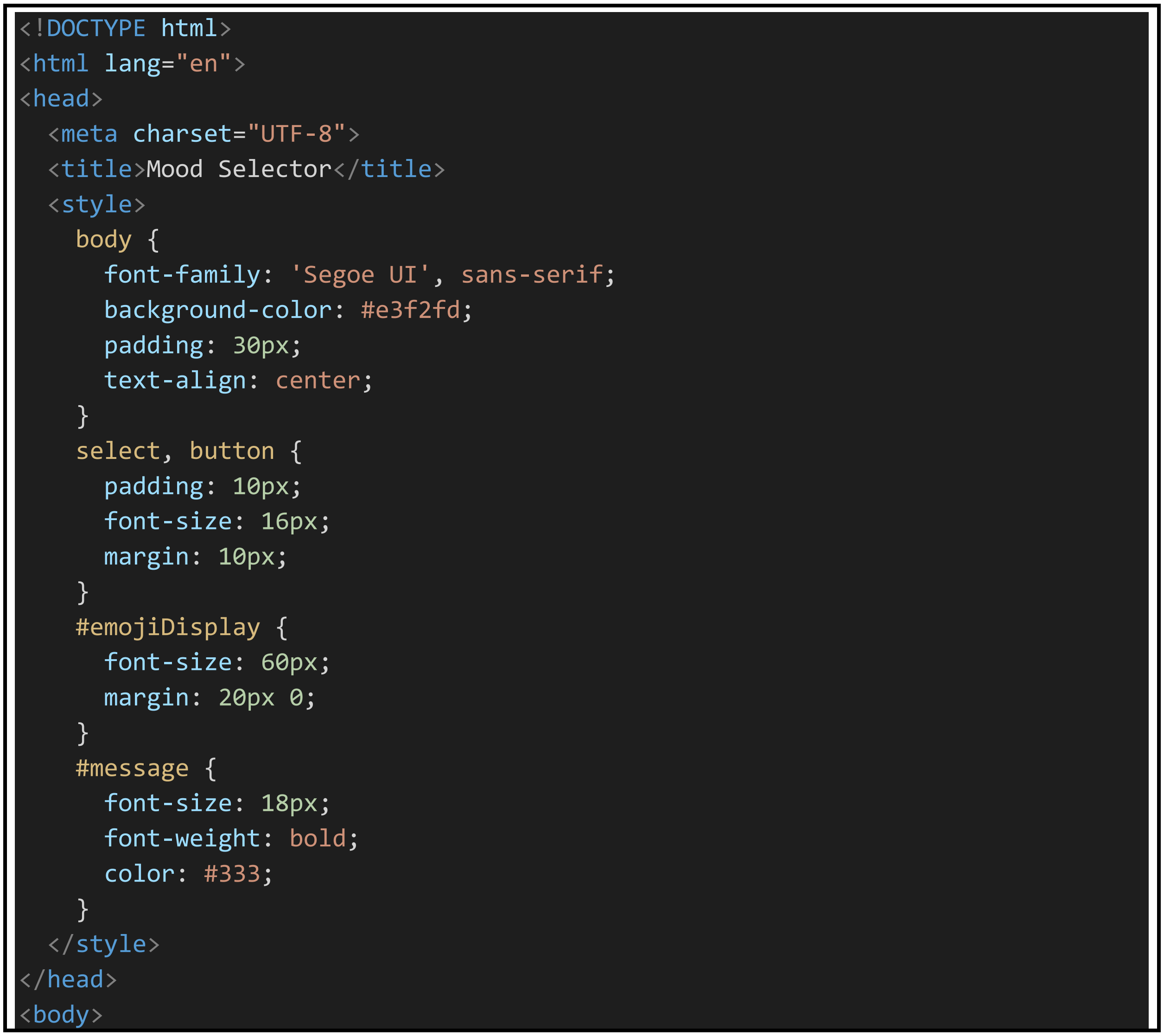
Practical Implementation: Mood Selector Webpage

A fun interactive page was created titled **“Mood Selector”**, allowing users to choose their mood, display related emojis, and receive a motivational message.

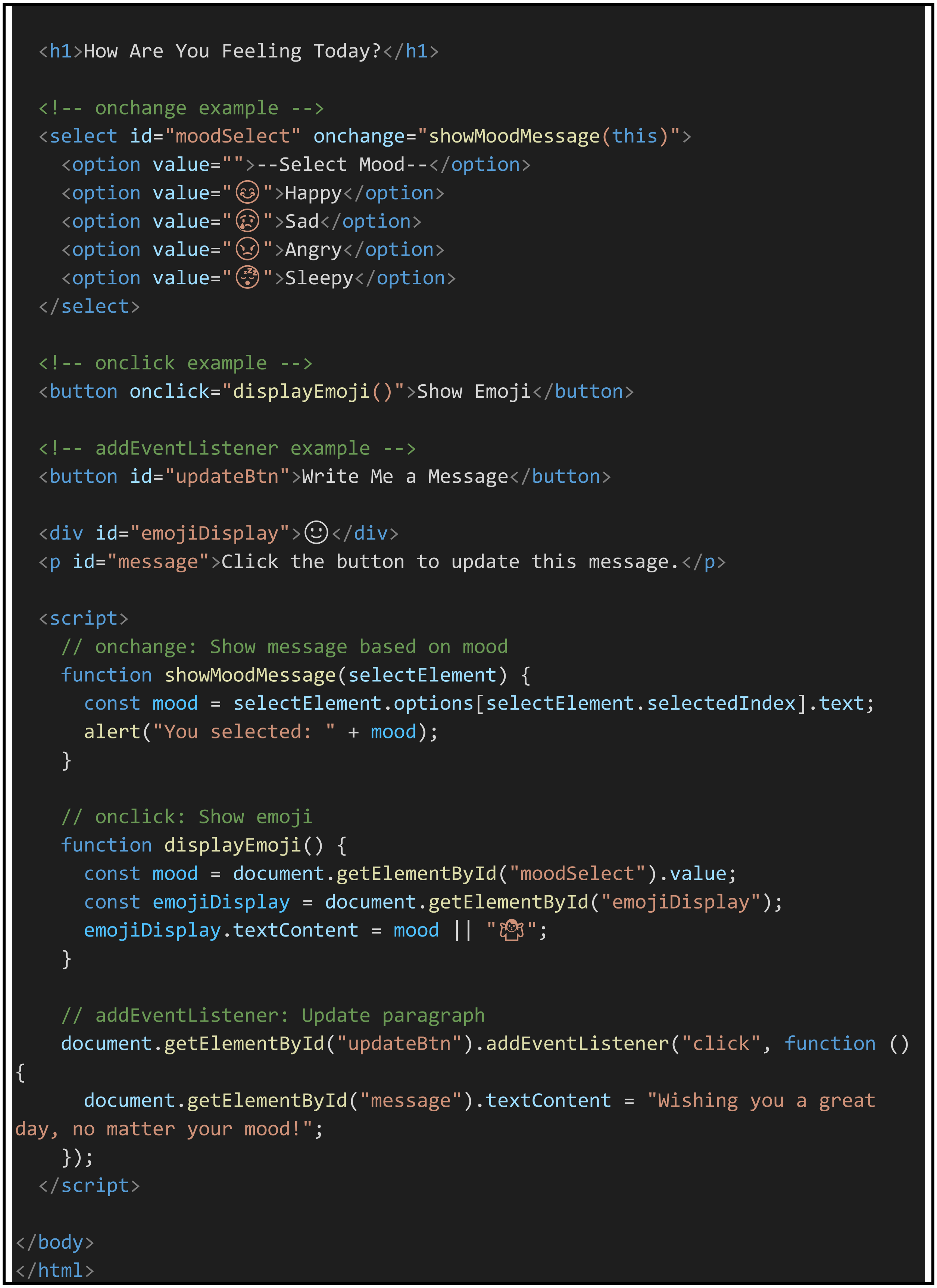
Features Developed:

Feature	Event Used	Description
Mood Dropdown	onchange	When the user selects a mood (e.g., Happy, Sad), an alert displays the selected mood.
Show Emoji Button	onclick	When clicked, shows an emoji based on the selected mood. If no mood is selected, a shrug emoji is shown.
Message Button	addEventListener	Updates a paragraph with a kind message when clicked.

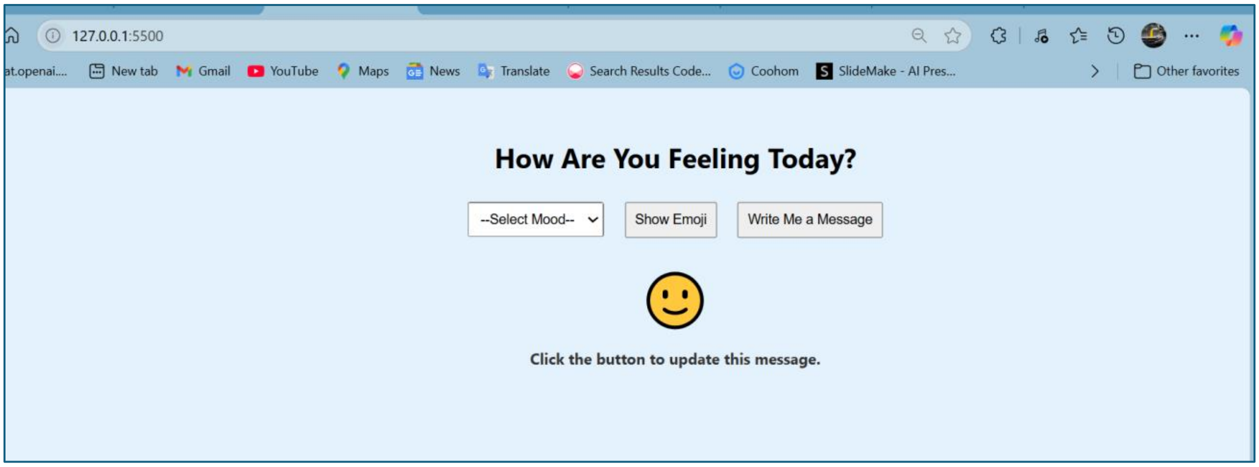
CODING



```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Mood Selector</title>
  <style>
    body {
      font-family: 'Segoe UI', sans-serif;
      background-color: #e3f2fd;
      padding: 30px;
      text-align: center;
    }
    select, button {
      padding: 10px;
      font-size: 16px;
      margin: 10px;
    }
    #emojiDisplay {
      font-size: 60px;
      margin: 20px 0;
    }
    #message {
      font-size: 18px;
      font-weight: bold;
      color: #333;
    }
  </style>
</head>
<body>
```



```
  <h1>How Are You Feeling Today?</h1>

  <!-- onchange example -->
  <select id="moodSelect" onchange="showMoodMessage(this)">
    <option value="">--Select Mood--</option>
    <option value="😊">Happy</option>
    <option value="😢">Sad</option>
    <option value="😠">Angry</option>
    <option value="😴">Sleepy</option>
  </select>

  <!-- onclick example -->
  <button onclick="displayEmoji()">Show Emoji</button>

  <!-- addEventListener example -->
  <button id="updateBtn">Write Me a Message</button>

  <div id="emojiDisplay">☺</div>
  <p id="message">Click the button to update this message.</p>

  <script>
    // onchange: Show message based on mood
    function showMoodMessage(selectElement) {
      const mood = selectElement.options[selectElement.selectedIndex].text;
      alert("You selected: " + mood);
    }

    // onclick: Show emoji
    function displayEmoji() {
      const mood = document.getElementById("moodSelect").value;
      const emojiDisplay = document.getElementById("emojiDisplay");
      emojiDisplay.textContent = mood || "🤷";
    }

    // addEventListener: Update paragraph
    document.getElementById("updateBtn").addEventListener("click", function ()
{
      document.getElementById("message").textContent = "Wishing you a great
day, no matter your mood!";
    });
  </script>

</body>
</html>
```

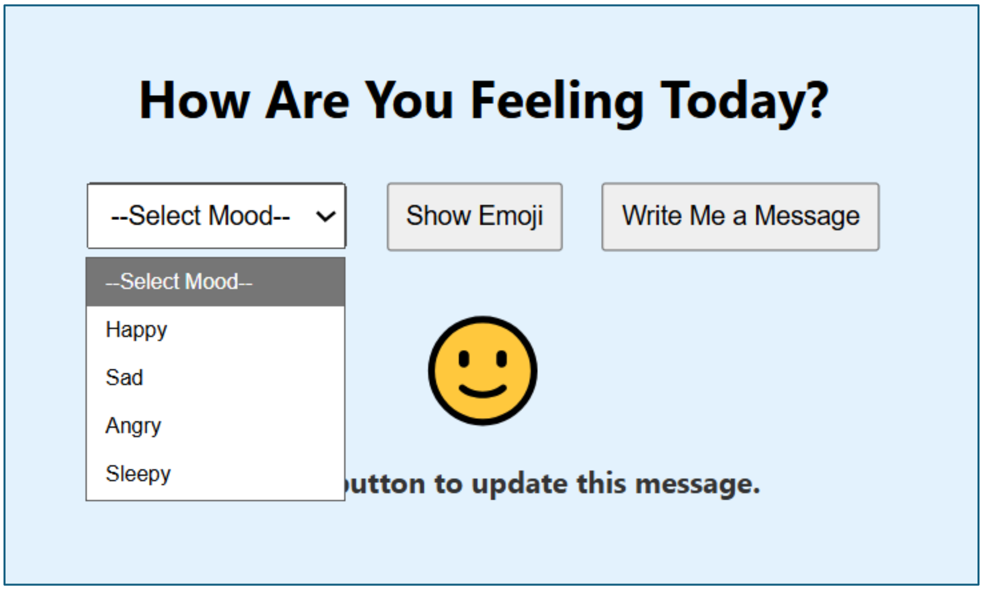


How Are You Feeling Today?

Click the button to update this message.

- **onchange:**

```
function showMoodMessage(selectElement) {
  const mood = selectElement.options[selectElement.selectedIndex].text;
  alert("You selected: " + mood);
}
```



How Are You Feeling Today?

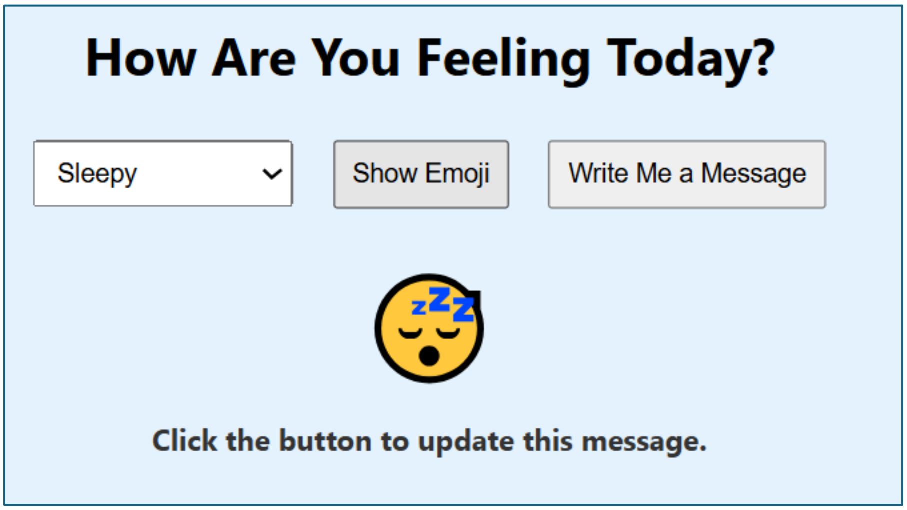
utton to update this message.

- **onclick:**

```
<button onclick="displayEmoji()">Show Emoji</button>
```

javascript

```
function displayEmoji() {
  const mood = document.getElementById("moodSelect").value;
  document.getElementById("emojiDisplay").textContent = mood || "🤷";
}
```



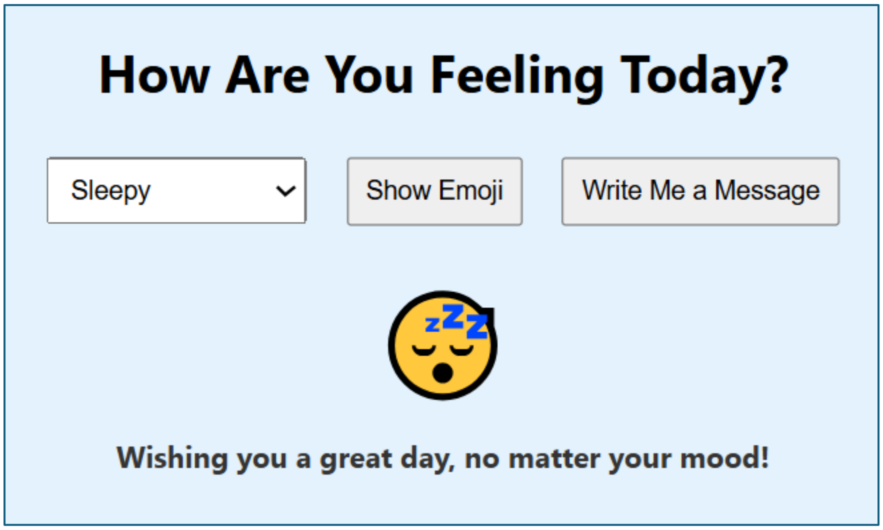
How Are You Feeling Today?

Click the button to update this message.

- **addEventListener:**

javascript

```
document.getElementById("updateBtn").addEventListener("click", function () {
  document.getElementById("message").textContent = "Wishing you a great day, no matter your mood!";
});
```



How Are You Feeling Today?

Wishing you a great day, no matter your mood!

CONCUSLION:

In today's session, I explored advanced JavaScript event handling concepts including onclick, onchange, and addEventListener(). To reinforce my understanding, I built a fully functional interactive project called **"Mood Selector"** using **Visual Studio Code (VS Code)**.

This practice helped me clearly understand how different events work in real-time user interactions. Using onclick, I triggered changes through button clicks; with onchange, I responded to user input from a dropdown; and through addEventListener, I attached dynamic event handling in a more modern and modular way.