NORAIZ RANA

Task #5

Topic

DOM Manipulation & Event Handling in Js

Date:

4 July 2025

SUBMITTED TO:

APPVERSE TECHNOLOGIES

Key Concepts Learned

1. What is DOM?

- **DOM (Document Object Model)** is a programming interface for HTML and XML documents.
- It represents the structure of a webpage as a tree of nodes (elements, attributes, text).
- JavaScript can interact with and change the DOM dynamically (add, remove, update elements).

2. Accessing DOM Elements

- Using built-in JavaScript methods like:
- `document.getElementById("id")`
- `document.getElementsByClassName("class")`
- `document.getElementsByTagName("tag")`
- `document.querySelector(".class, #id, tag")`
- `document.querySelectorAll("selector")`
- Example:
- `const heading = document.getElementById("main-title");`

3. Manipulating DOM Elements

- **Changing text or HTML:**
- `element.textContent = "New Text";`
- `element.innerHTML = "<strong>Bold Text</strong>";`
- **Changing styles:**
- `element.style.color = "red";`
- `element.style.backgroundColor = "yellow";`
- **Adding/removing classes:**
- `element.classList.add("highlight");`
- `element.classList.remove("hidden");`
- `element.classList.toggle("active");`
- **Creating new elements:**
- `const newDiv = document.createElement("div");`
- `newDiv.textContent = "I am new!";`
- `document.body.appendChild(newDiv);`

4. Event Handling in JavaScript

- Events are actions performed by users (click, submit, hover, keypress, etc.).
- You can **listen** for events using:
- `element.addEventListener("event", function);`
- Common Events:
    - `click`, `mouseover`, `mouseout`
    - `keydown`, `keyup`
    - `submit`, `change`, `input`

- Example:

```
const btn = document.getElementById("myBtn");
btn.addEventListener("click", function () {
  alert("Button clicked!");
});
```

5. Event Object and Event Target

- The event listener receives an `event` object with useful data.

```
element.addEventListener("click", function (e) {
  console.log(e.target); // Element that triggered the event
});
```

6. Event Delegation

- Technique to handle events on parent elements instead of each child.
- Improves performance and makes code cleaner:

```
document.getElementById("list").addEventListener("click", function (e)
{
  if (e.target.tagName === "LI") {
    e.target.style.color = "blue";
  }
});
```

Hands-on Practice

- Made buttons change background color dynamically.
- Created modal window with open/close event handling.

Challenges Faced

- Initially struggled with nested selectors and event delegation.
- Handling dynamically added elements required understanding event bubbling.

Learning DOM manipulation and event handling gave me control over the structure and behavior of web pages. It is a crucial part of frontend development and forms the foundation for frameworks like React and Vue. This knowledge will directly support my ability to build interactive and dynamic web applications.