## 🧩 Task 6: Events and Event Delegation.

---

## ✅ Task 6: Events & Event Delegation

### 🎯 Goals:

- Understand how JavaScript handles user interactions
- Use addEventListener effectively
- Master event objects (e)
- Learn **event delegation** — a crucial technique used in dynamic interfaces (especially React)

---

## 📘 What Are Events?

Events are notifications that something has happened — like a user clicking a button, typing into a form, or moving their mouse.

JavaScript lets you **listen** for these events and **run code** in response.

---

## 🔍 Key Concepts & Examples

---

### 🔸 1. Adding an Event Listener

```
const btn = document.querySelector("#clickMe");


btn.addEventListener("click", function () {
  console.log("Button clicked!");
});
```

You can also use arrow functions:

```
btn.addEventListener("click", () => {
  console.log("Button clicked!");
```

});

---

### 🔸 2. The Event Object (e)

Every event handler receives a special object that gives info about the event:

```
document.addEventListener("click", function (e) {
  console.log(e.target); // The exact element that was clicked
  console.log(e.type);   // e.g., "click"
});
```

---

### 🔸 3. Common Event Types

| Event | Trigger |
|---|---|
| click | Clicking an element |
| input | Typing in an input field |
| submit | Submitting a form |
| keydown | Pressing a keyboard key |
| mouseover | Moving cursor over an element |
| change | Changing value in <select> or checkbox |

---

### 🔸 4. Removing Event Listeners

```
function handleClick() {
  alert("Clicked!");
  button.removeEventListener("click", handleClick);
}

const button = document.querySelector("button");
```

button.addEventListener("click", handleClick);

---

### 🔸 5. Event Delegation ( 🔥 Very Important)

Event delegation allows you to **listen on a parent**, and handle events for dynamically added child elements.

**Why?**

You can't add listeners to elements that don't exist yet. Instead, attach the listener to a parent and detect the clicked child.

**Example:**

```
const list = document.querySelector("ul");


list.addEventListener("click", function (e) {
 if (e.target.tagName === "LI") {
  e.target.classList.toggle("completed");
 }
});
```

✅ e.target tells you which exact element was clicked
✅ Use if to match specific elements

---

### 🧪 Practice Exercises

### ✅ Exercise 1: Click Counter

- Create a button and a  showing a count
- When clicked, increase the number by 1

---

### ✅ Exercise 2: Dynamic List with Delete (using delegation)

- Create a <ul> and a form to add <li> items

- Each <li> should have a “ ❌ ” delete button
- Use **event delegation** on the <ul> to handle delete clicks

---

### ✅ Exercise 3: Input Preview

- Create a form with a name field
- As the user types, display a live preview (e.g., “Hi, Ali!”)

---

### 💬 Reflection Questions

1. What is the purpose of the event object (e)?
2. Why is event delegation important for dynamic apps?
3. What would happen if you tried to attach an event listener to an element that doesn’t exist yet?