AMALITECH

**Week 2.2**

**Lab Activity**: Building with Closures and ***this*** keyword

**Learning Objectives:**

- Understand the concept of closures and how functions maintain access to their lexical scope.
- Grasp the dynamic nature of ***this*** keyword in JavaScript.
- Learn how closures can be used to manage and control the value of `this` in different contexts.
- Apply these concepts to build reusable and modular components in JavaScript.

**Tasks**:

1. Object Methods and ***this:***
   - Create a `Person` object with the following properties:
     - `name` (string)
     - `age` (number)
   - Add a method `greet()` to the `Person` object that logs a message like "Hello, my name is [name] and I'm [age] years old."
   - Experiment with calling `greet()` directly on the `Person` object, using `call()`, `apply()`, and `bind()`. Observe how the value of `this` changes in each context.
2. Event Handlers and ***this***:
   - Create a button element in HTML.
   - Attach an event listener to the button that calls a function `handleClick()`.
   - Inside `handleClick()`, try to log the properties of the button (e.g., `this.id`, `this.textContent`). Observe the value of this.
   - Modify the code to use an arrow function for the event listener. Notice the difference in how `this` behaves.
3. Private Data with Closures and `this`:
   - Create a function `createCounter()` that:
     - Has a private variable `count` initialized to 0.
     - Returns an object with two methods:

        1. `increment()`: Increments the `count` and logs the new value to the console using `this.count`.
        2. `getCount()`: Returns the current value of `count`.

4. Reusable Component with Closure and `this`.
    - Create a function `createTimer(duration, elementId)` that:
        - Takes a `duration` in seconds and an `elementId` as input.
        - Starts a timer that counts down from `duration` to 0.
        - Updates the content of the element with the given `elementId` to display the remaining time every second.
        - When the timer reaches 0, logs a message to the console.
        - Uses closures to store the timer's state (remaining time) and `this` to refer to the correct element.

**Evaluation:**

- Code Functionality:
    - Ensure that all tasks are completed and functions work as expected.
    - Verify that you're able to control the value of `this` in different contexts using closures, `call()`, `apply()`, and `bind()`.
- Understanding:
    - Demonstrate a clear understanding of closures and the dynamic nature of `this`.
    - Explain how closures can be used to capture the value of `this` and make it available in different parts of your code.
    - Discuss the benefits of using closures and `this` to create reusable components.