# TRAINING DAY- 7

27th June 2025

## TOPICS COVERED – JavaScript Functions & Scope

### Function Calling & Return Function

We revised how to define and call JavaScript functions.

A **function** can return values using the return keyword.

**Example:**

```
function add(a, b) {
  return a + b;
}
console.log(add(2, 3)); // Output: 5
```

### Block Scope: var vs let

- var has **function scope**: it's visible throughout the function.
- let has **block scope**: it's only visible inside the {} block where it's defined.

```
if (true) {
  var x = 10;
  let y = 20;
}
console.log(x); // 10
console.log(y); // Error: y is not defined
```

## Arrow Functions

Arrow functions offer a cleaner syntax and don't bind their own this.

```
const multiply = (a, b) => a * b;
console.log(multiply(3, 4)); // 12
```

## Higher-Order Functions

These are functions that take other functions as arguments or return functions.

```
function greet(callback) {
  callback("Good morning");
}

function showMessage(message) {
  console.log("Message:", message);
}

greet(showMessage); // Output: Message: Good morning
```

## Callback Functions

A callback is a function passed into another function to be called later.

```
setTimeout(() => {
  console.log("Executed after 2 seconds");
}, 2000);
```

## CODE TASKS

1. Create an arrow function that adds two numbers.
2. Write a higher-order function that takes a callback.
3. Demonstrate var and let with examples inside a loop.
4. Use setTimeout() with a callback.