**IIFE**

IIFE is a javascript function that is invoked and executed. Variables defined within the IIFE cannot be accessible by the outside world, preventing polluting of the global scope. So the main purpose to use IIFE is to run code quickly and ensure data privacy.

**Callbacks**

A callback in javascript is essentially a function that is supplied as a parameter to another function and then called or run inside that function. In this case, a function must wait for another function to run or return a value, forming a functional chain. This is why callback is commonly utilised in javascript’s asynchronous operations to offer synchronous functionality.

**Scope**

Variable access is what scope refers to. When a programme is executing, what variables do I have access to? You’re always in root scope in javascript, i.e. the window scope. Variables, functions, and objects are all contained within the scope, which is just a box with a border. These limits impose constraints on variables and affect whether or not you have access to them. It restricts a variable’s visibility or availability to other portions of the code. This notion must be understood thoroughly since it aids in the separation of logic in your code and enhances usability.

**Closures**

A closure is a function that executes inside of another function and has access to the variables of the outer function. While this definition looks to be simple, the scope is where the real magic occurs. The inner function has access to all variables defined in its scope, the scope of its parent function, and the global variables. It’s important to note that at this time, the outer function can’t access the inner function variable.

**Hoisting**

When developers are unfamiliar with the idea of hoisting in javascript, they sometimes receive unexpected outcomes. You may call a function in javascript before it is declared and not get the ‘Uncaught ReferenceError’ error. The reason for this is that before executing the code, the javascript interpreter automatically pushes the variables and function declarations to the top of the current scope.