Lesson 05 Demo 01

Working with IIFEs Callbacks and Closures

Objective: To demonstrate the process of implementing Immediately Invoked Function Expressions (IIFEs), callbacks, and closures in JavaScript for enhancing modularity, encapsulation, and asynchronous execution

Tools required: Visual Studio Code

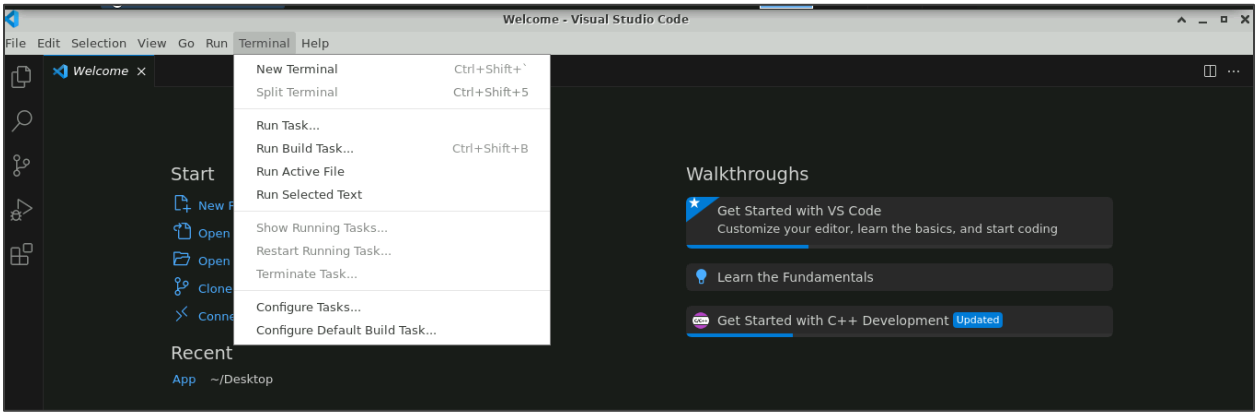
Prerequisites: None

Steps to be followed:

1. Write a JavaScript program with IIFEs, callbacks, and closures
2. Test and verify the IIFEs, callbacks, and closures in action

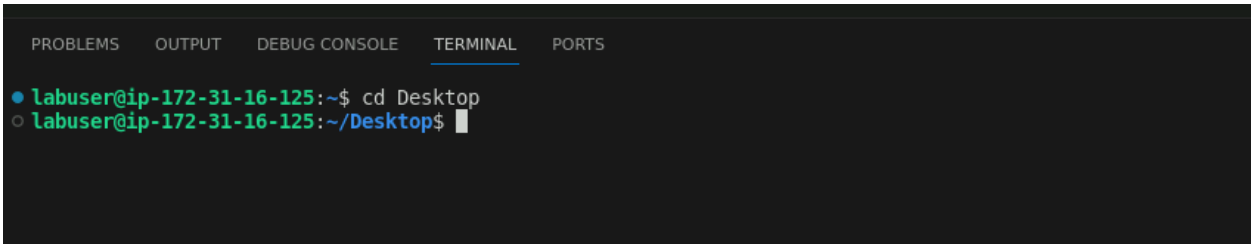
Step 1: Write a JavaScript program with IIFEs, callbacks, and closures

1.1 Open Visual Studio Code and navigate to the terminal to create an **src** folder



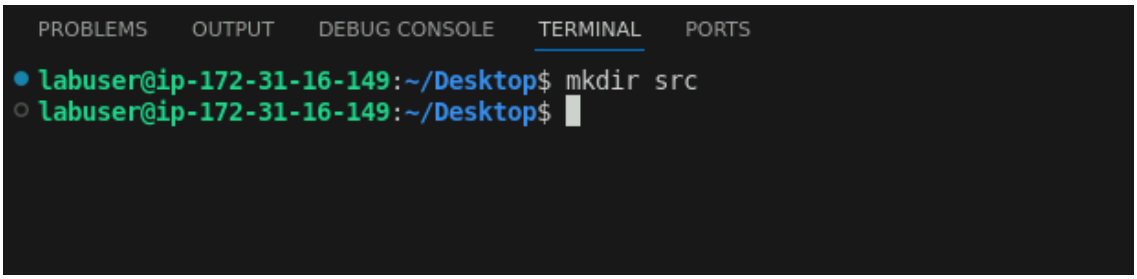
1.2 Execute the following command to navigate to the desktop directory:

cd Desktop



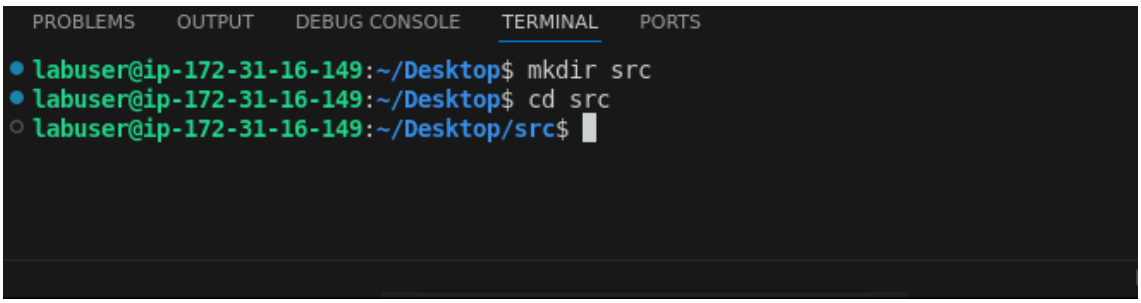
1.3 Run the following command to create a folder named **src**:

mkdir src

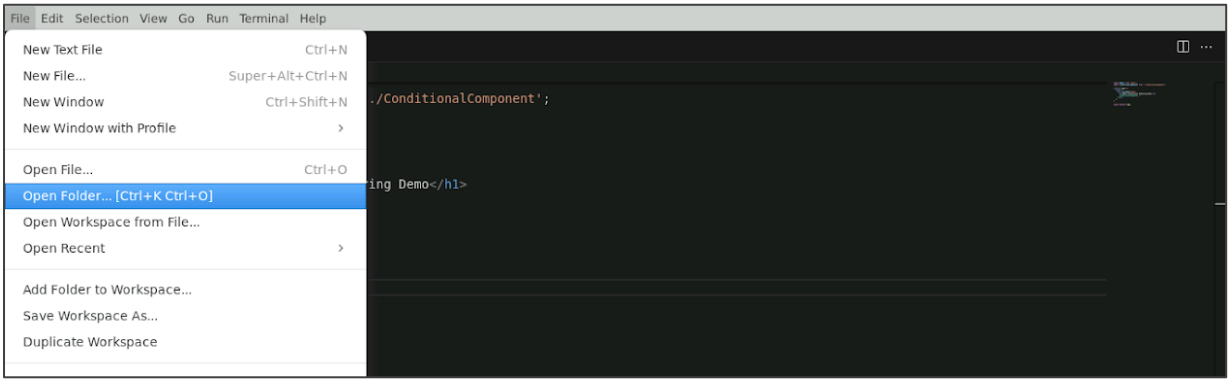


1.4 Execute the following command to navigate to the **src** directory:

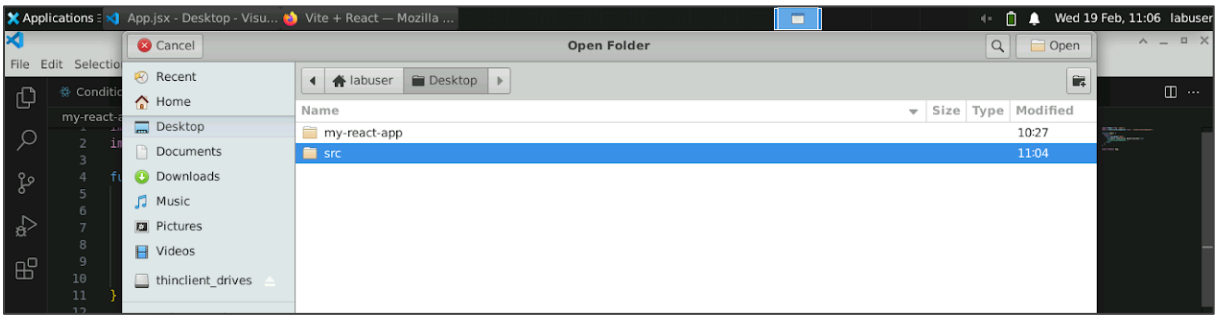
cd src

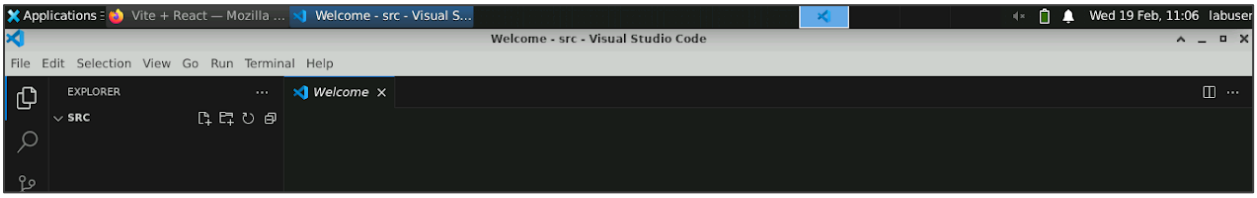


1.5 Click on **File** and open the folder

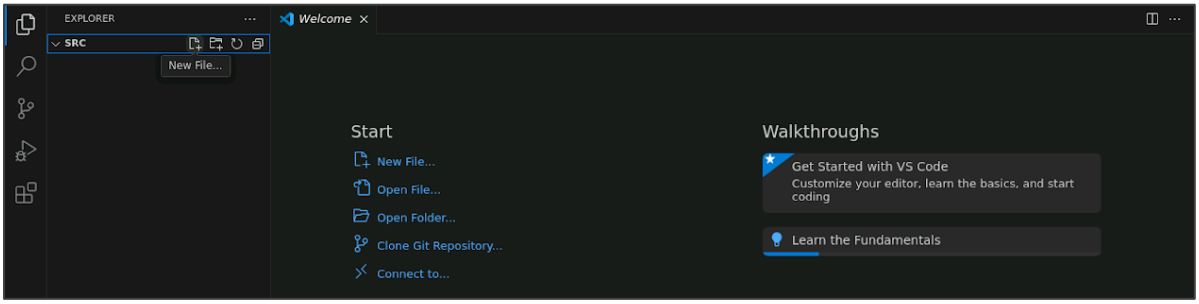


1.6 Open the **src** folder



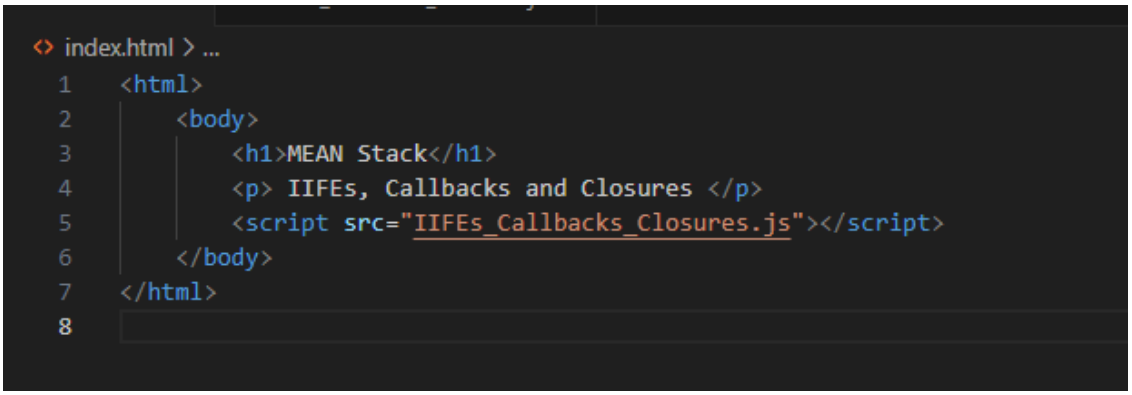


1.7 Click on the **src** folder of the project, select the **New File...** option, and enter the filename as **index.html**



1.8 Write the given code in **index.html**:

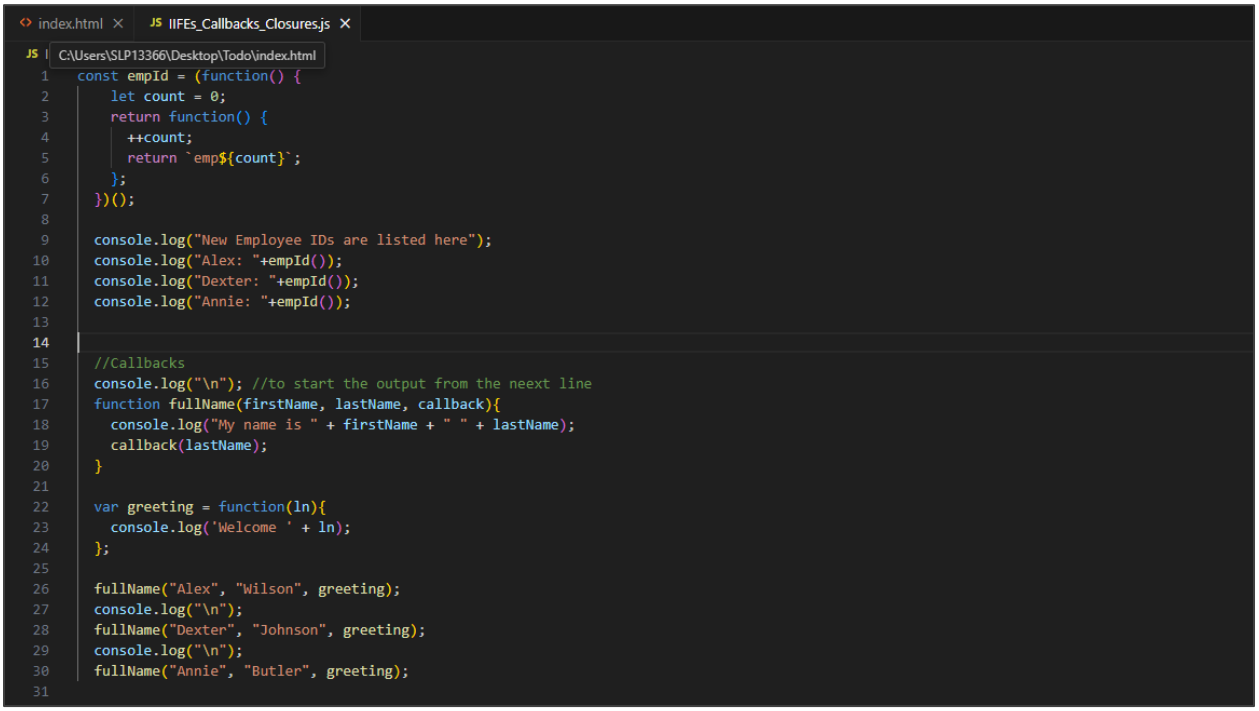
```
<html>
  <body>
    <h1>MEAN Stack</h1>
    <p> IIFEs, Callbacks and Closures </p>
    <script src="IIFEs_Callbacks_Closures.js"></script>
  </body>
</html>
```



1.9 Click on the **src** folder of the project, select the **New File**... option, enter the filename as **IIFEs_Callbacks_Closures.js,** and write the code shown below:

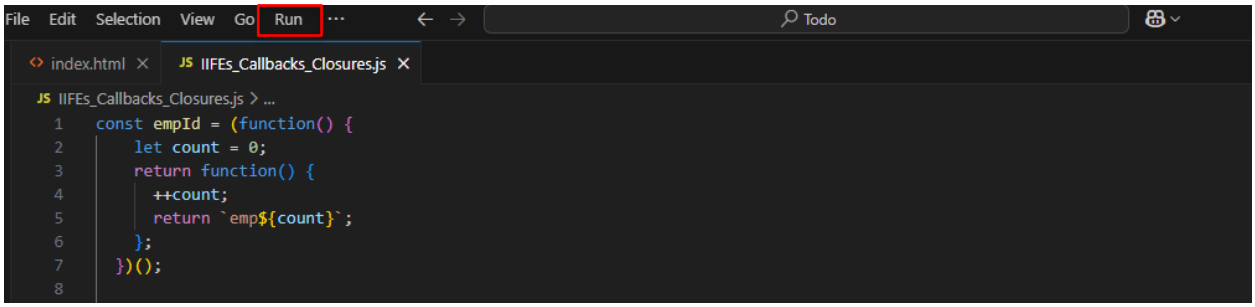
```
const empId = (function() {
  let count = 0;
  return function() {
   ++count;
   return `emp${count}`;
  };
 })();
 console.log("New Employee IDs are listed here");
 console.log("Alex: "+empId());
 console.log("Dexter: "+empId());
 console.log("Annie: "+empId());
 //Callbacks
 console.log("\n"); //to start the output from the next line
 function fullName(firstName, lastName, callback){
  console.log("My name is " + firstName + " " + lastName);
```

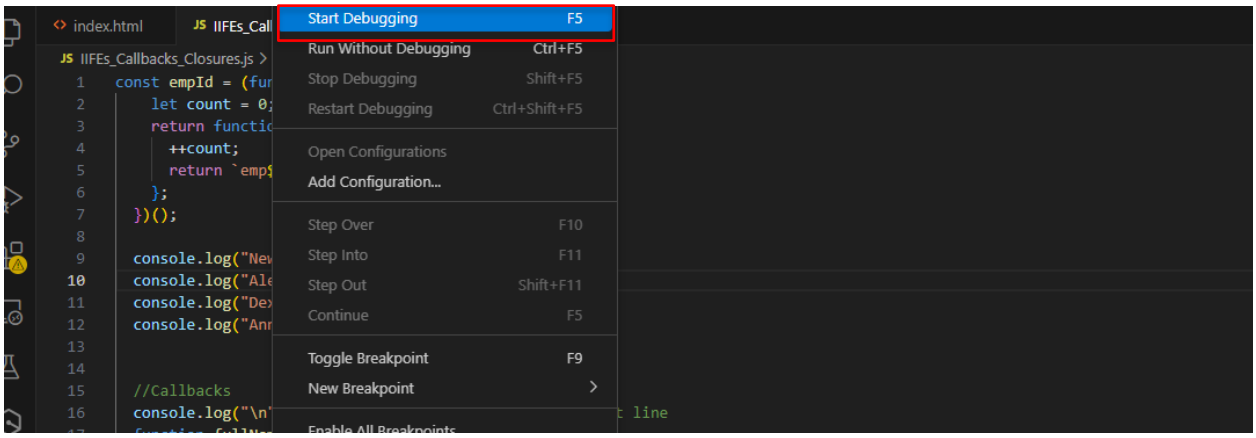
```
 callback(lastName);
}
var greeting = function(ln){
 console.log('Welcome ' + ln);
};
fullName("Alex", "Wilson", greeting);
console.log("\n");
fullName("Dexter", "Johnson", greeting);
console.log("\n");
fullName("Annie", "Butler", greeting);
```



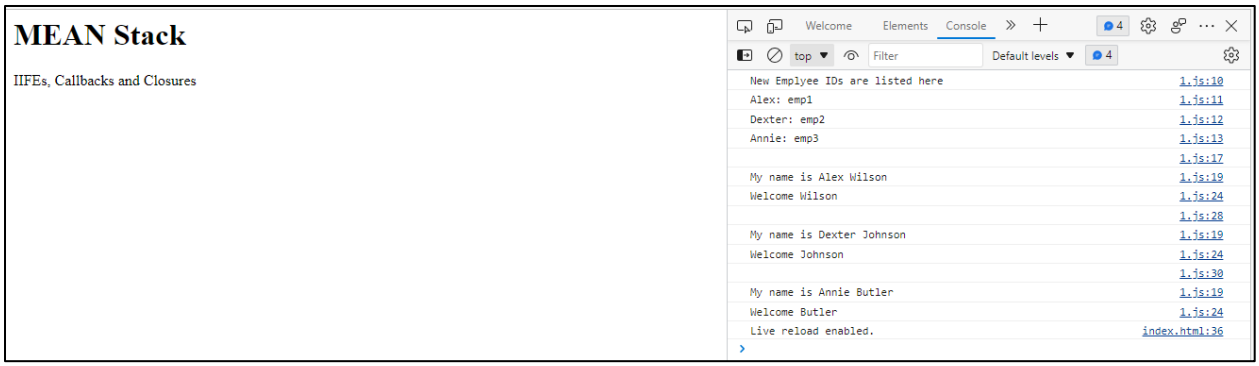
Step 2: Test and verify the IIFEs, callbacks, and closures in action

2.1 Click on **Run** and then on **Start Debugging** to execute the JavaScript file





2.2 When the server starts running, right-click and select the **Inspect Element** option and click on the **Console** tab



By following the above steps, you have successfully implemented IIFEs, callbacks, and closures in JavaScript, demonstrating how they help in creating modular, encapsulated,

and efficient code execution with controlled variable scope and asynchronous behavior.