

## Lesson 11 Demo 07

### Implementing Mocha Hooks for Function Testing

**Objective:** To install Mocha and verify its hook functions for organized and controlled function testing in a Node.js environment

**Tools Required:** Visual Studio

**Prerequisites:** Knowledge of JavaScript and Node.js


Steps to be followed:

1. Create the node app.js file called the simple function
2. Create a test file to check the above function with the hook concept

#### Step 1: Create the node app.js file called the simple function

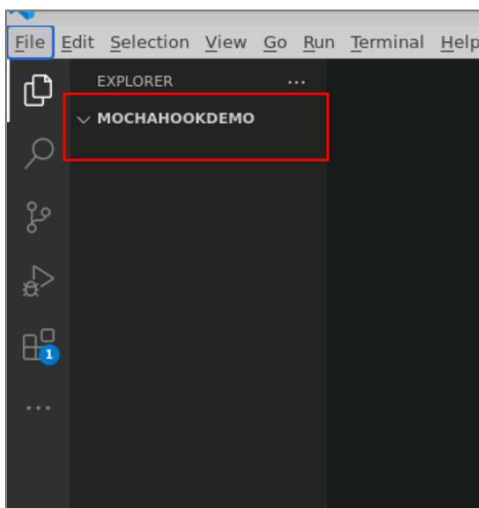
- 1.1 Run the below code to create a working directory to hold the application:

```
mkdir mochahookdemo  
cd mochahookdemo
```



```
labuser@ubuntu2204:~/Desktop$ mkdir mochahookdemo  
labuser@ubuntu2204:~/Desktop$ cd mochahookdemo/  
labuser@ubuntu2204:~/Desktop/mochahookdemo$
```

- 1.2 Open the folder in VS Code IDE



1.3 Open the terminal and run the below command to create a **package.json** file for the application

**npm init -y**

```
labuser@ubuntu2204:~/Desktop/mochahookdemo$ npm init -y
Wrote to /home/labuser/Desktop/mochahookdemo/package.json:

{
  "name": "mochahookdemo",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}

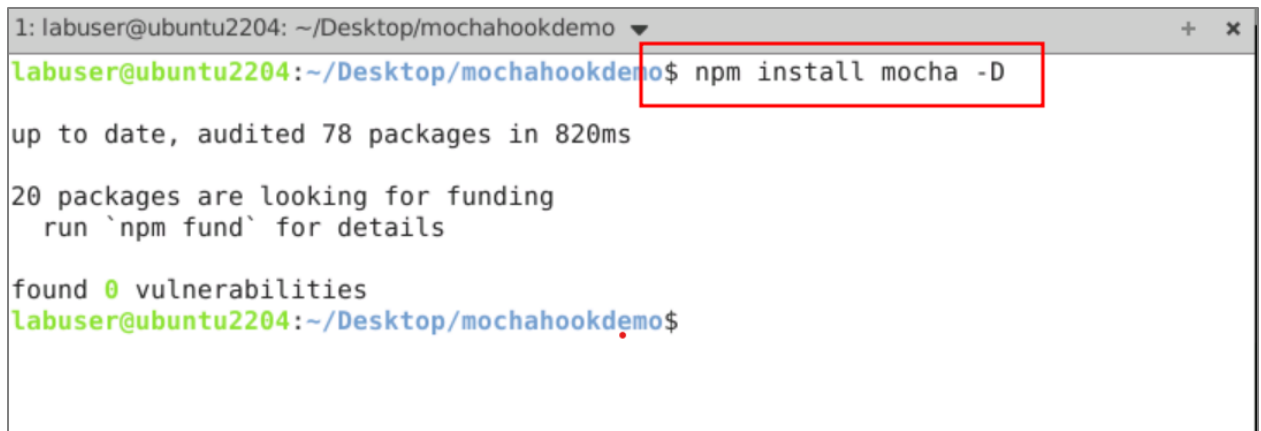
labuser@ubuntu2204:~/Desktop/mochahookdemo$
```

It creates a package.json file with a default option.

You can now view the package.json file.

```
{ } package.json x
{ } package.json > ...
1  {
2    "name": "nodejstestingmocha",
3    "version": "1.0.0",
4    "description": "",
5    "main": "index.js",
6    "scripts": {
7      "test": "echo \"Error: no test specified\" && exit 1"
8    },
9    "author": "",
10   "license": "ISC"
11 }
12
```

#### 1.4 Execute `npm install mocha -D` command



```
1: labuser@ubuntu2204: ~/Desktop/mochahookdemo
labuser@ubuntu2204:~/Desktop/mochahookdemo$ npm install mocha -D

up to date, audited 78 packages in 820ms

20 packages are looking for funding
  run `npm fund` for details

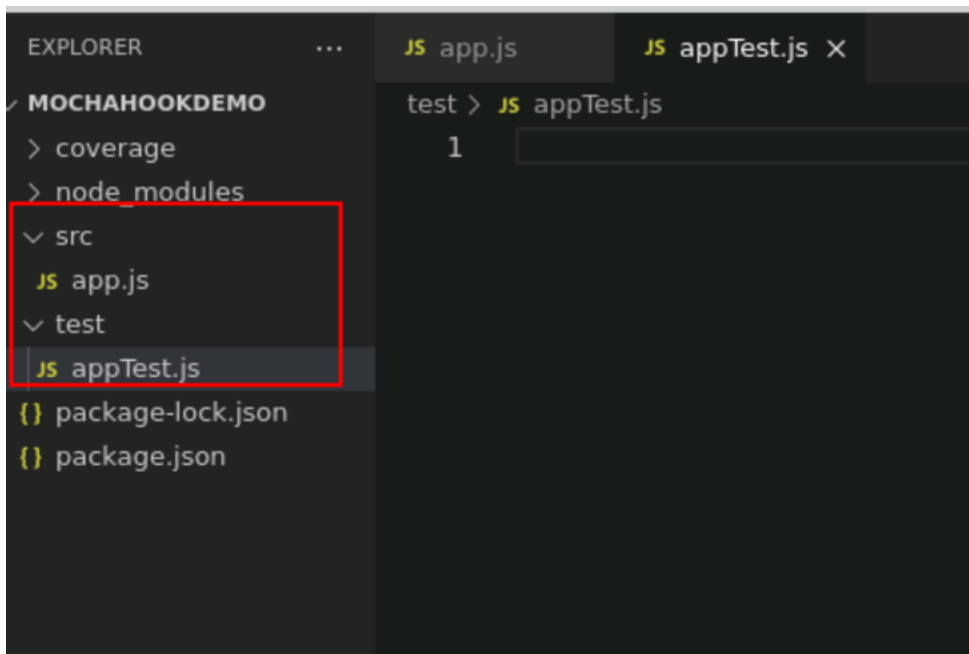
found 0 vulnerabilities
labuser@ubuntu2204:~/Desktop/mochahookdemo$
```

### Step 2: Create a test file to check the above function with the hook concept

2.1 Create a src folder and create an app.js file inside it

2.2 Create a test folder and create an appTest.js file inside it

**Note:** Refer to the following Project structure:

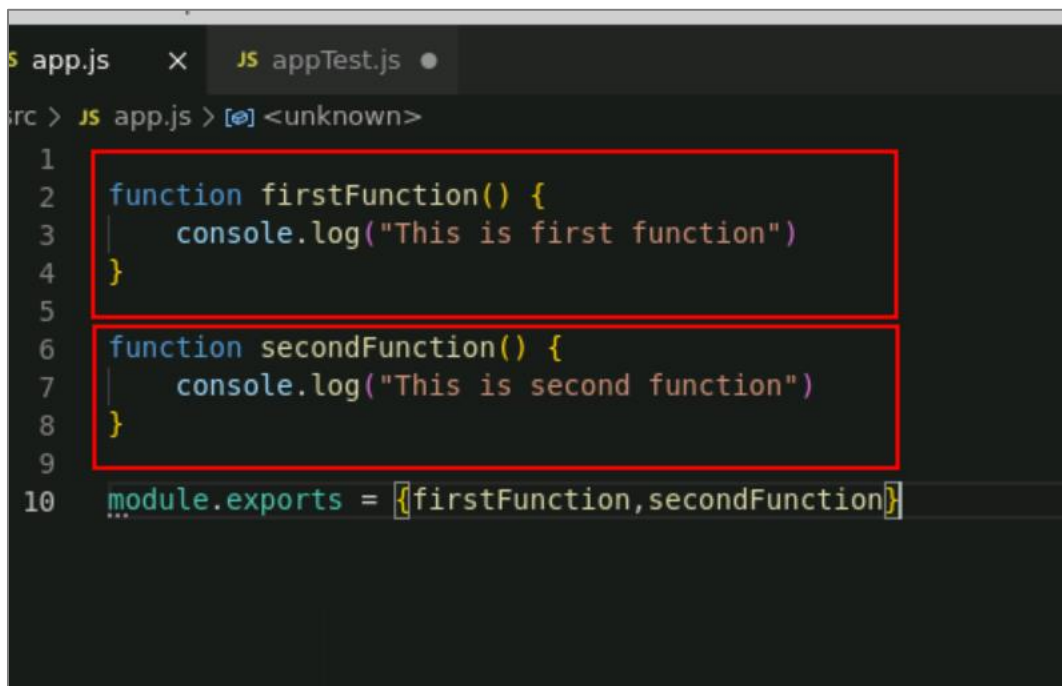


### 2.3 Write the below code in the app.js file

```
function firstFunction() {  
  console.log("This is first function")  
}
```

```
function secondFunction() {  
  console.log("This is second function")  
}
```

```
module.exports = { firstFunction, secondFunction }
```



```
src > JS app.js > [e] <unknown>  
1  function firstFunction() {  
2    console.log("This is first function")  
3  }  
4  
5  
6  function secondFunction() {  
7    console.log("This is second function")  
8  }  
9  
10 module.exports = {firstFunction,secondFunction}
```

### 2.4 Write the below code in the appTest.js file

```
const assert = require("assert");
```

```
const appRef = require("../src/app");
```

```
describe("Mocha hook Example", () => {  
  before(() => {  
    console.log("this function call before all it - only once");  
  });
```

```
  beforeEach(() => {  
    console.log("this function call before each it - again and again");  
  });
```

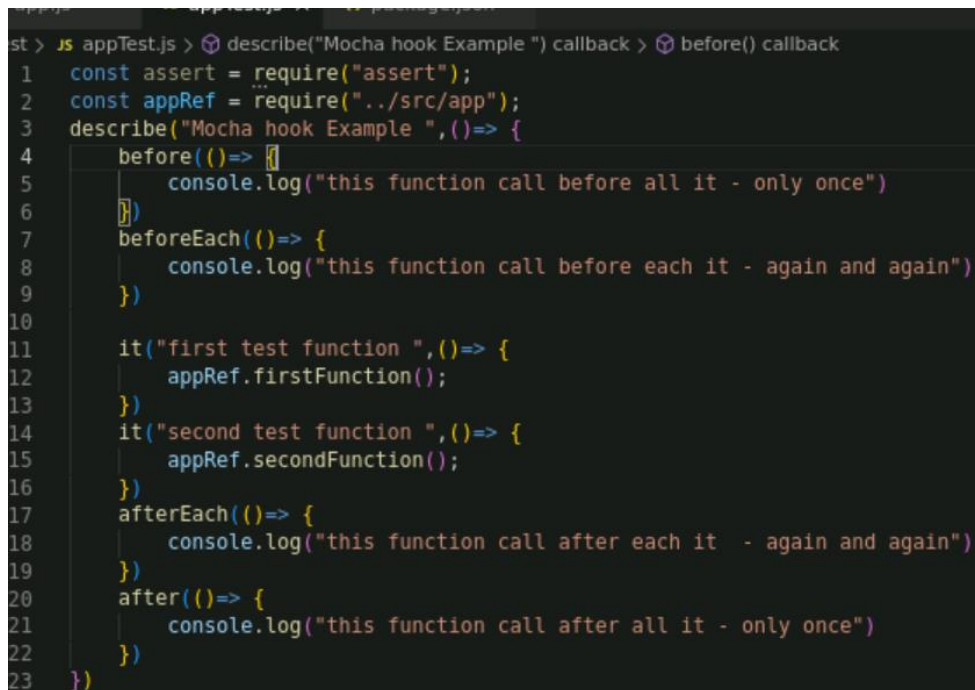
```
  it("first test function", () => {
```

```
    appRef.firstFunction();
  });
```

```
it("second test function", () => {
  appRef.secondFunction();
});
```

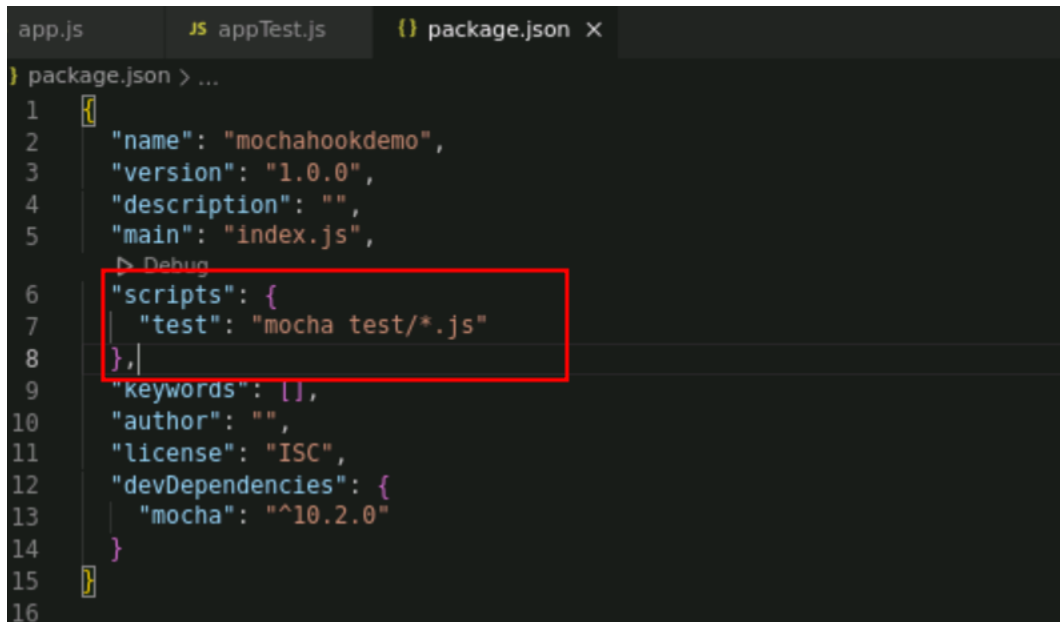
```
afterEach(() => {
  console.log("this function call after each it - again and again");
});
```

```
after(() => {
  console.log("this function call after all it - only one");
});
});
```



```
st > .js appTest.js > describe("Mocha hook Example ") callback > before() callback
1  const assert = require("assert");
2  const appRef = require("../src/app");
3  describe("Mocha hook Example ", () => {
4    before(() => {
5      console.log("this function call before all it - only once")
6    })
7    beforeEach(() => {
8      console.log("this function call before each it - again and again")
9    })
10
11    it("first test function ", () => {
12      appRef.firstFunction();
13    })
14    it("second test function ", () => {
15      appRef.secondFunction();
16    })
17    afterEach(() => {
18      console.log("this function call after each it - again and again")
19    })
20    after(() => {
21      console.log("this function call after all it - only once")
22    })
23  })
```

## 2.5 Write testing file details in the package.json file

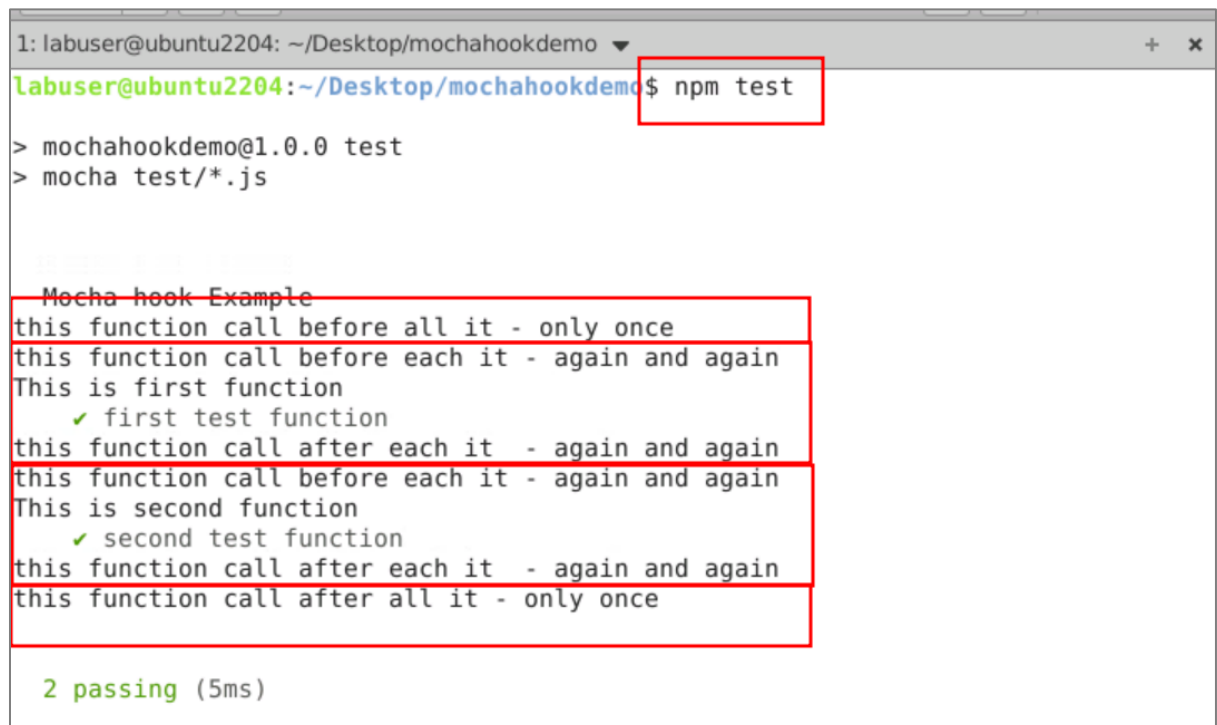


```

package.json > ...
1  {
2    "name": "mochahookdemo",
3    "version": "1.0.0",
4    "description": "",
5    "main": "index.js",
6    "scripts": {
7      "test": "mocha test/*.js"
8    },
9    "keywords": [],
10   "author": "",
11   "license": "ISC",
12   "devDependencies": {
13     "mocha": "^10.2.0"
14   }
15 }
16

```

## 2.6 Open the terminal and test the code



```

1: labuser@ubuntu2204: ~/Desktop/mochahookdemo
labuser@ubuntu2204:~/Desktop/mochahookdemo$ npm test

> mochahookdemo@1.0.0 test
> mocha test/*.js

Mocha hook Example
this function call before all it - only once
this function call before each it - again and again
This is first function
  ✓ first test function
this function call after each it - again and again
this function call before each it - again and again
This is second function
  ✓ second test function
this function call after each it - again and again
this function call after all it - only once

2 passing (5ms)

```

By following these steps, you have successfully implemented Mocha hooks for function testing in a Node.js environment to enhance test suite organization and execution control.