To create a JavaScript project and implement maps and classes, you can follow these steps:

1. Open your IDE and create a new folder for your project. Name it something like "maps-classes-project".

2. Inside the project folder, create two files: "index.html" and "script.js".

3. Open "index.html" in your IDE and add the following basic HTML structure:

```
<!DOCTYPE html>
<html>
<head>
    <title>Maps and Classes</title>
    <script src="script3.js"></script>
</head>
<body>
    <h1>JavaScript Maps and Classes</h1>
</body>
</html>
```

4. In the same project folder, open "script.js" in your IDE and write the JavaScript code to implement maps and classes. Here's an example:

```
// Define a class
class Employee {
    constructor(empname, empid, city) {
        this.empname = empname;
        this.empid = empid;
        this.city = city;
    }
    displayDetails() {
        console.log('Employee Name:', this.empname);
        console.log('Employee ID:', this.empid);
        console.log('City:', this.city);
        console.log('----------------------');
    }
}
// Create a Map and add employee details to it
const employeesMap = new Map();
```

```
const employee1 = new Employee('John Doe', 'EMP001', 'New York');
const employee2 = new Employee('Jane Smith', 'EMP002', 'San Francisco');
employeesMap.set('EMP001', employee1);
employeesMap.set('EMP002', employee2);
// Access employee details using Map   (// const empIdToFind = 'EMP002'; (1st case) // //
const empIdToFind = 'EMP001'; (2nd case) //)
const empIdToFind = 'EMP004';
if (employeesMap.has(empIdToFind)) {
   const empDetails = employeesMap.get(empIdToFind);
   empDetails.displayDetails();
} else {
   console.log('Employee with ID', empIdToFind, 'not found.');
}
```

In the above code, we define a class `Employee` with a constructor to initialize employee details (name, ID, and city) and a method `displayDetails` to log the employee details to the console.

We then create a `Map` named `employeesMap` and add two employee objects to it using their respective employee IDs as keys.

Next, we demonstrate how to access employee details from the map using an employee ID (`EMP001`) and display the details using the `displayDetails` method.

5. Save both files.

6. Open "index.html" in a web browser, and you should see a blank page.

7. Open the browser's developer tools (usually accessible through right-clicking on the page and selecting "Inspect" or "Inspect Element").

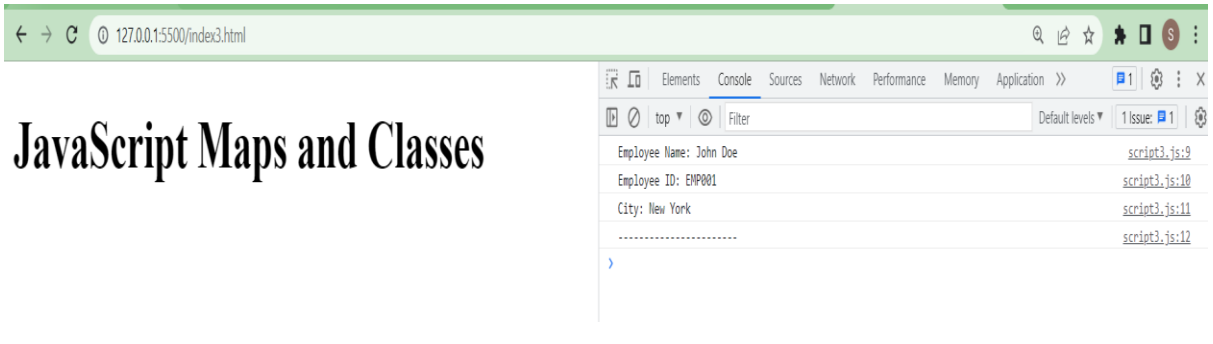
8. Switch to the "Console" tab in the developer tools.

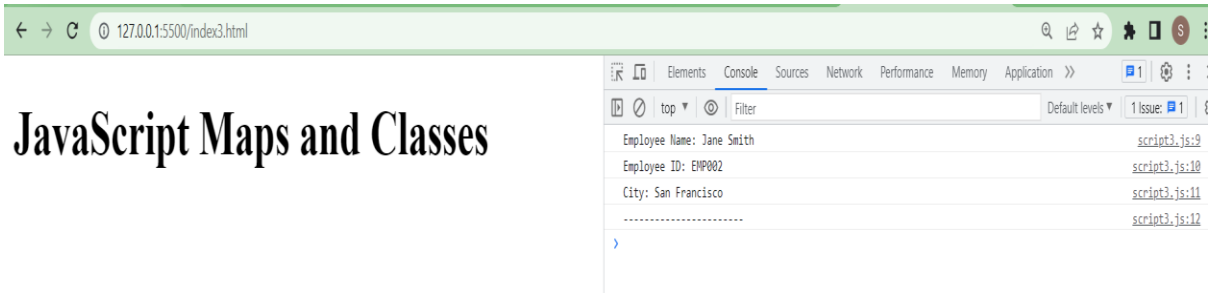
9. Reload the "index.html" page.

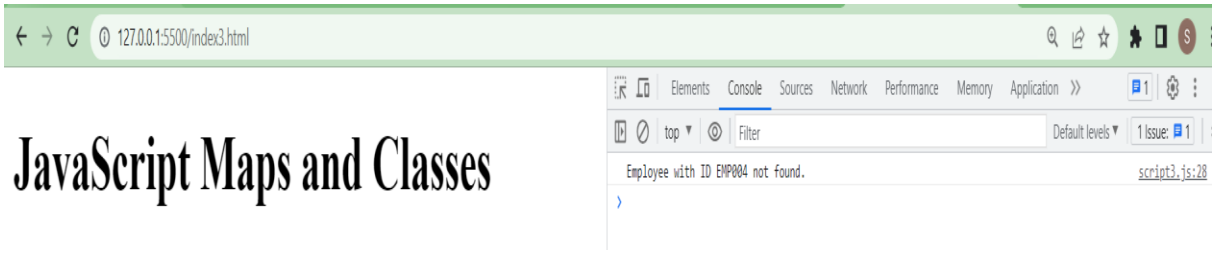
10. In the console, you should see the employee details for the employee with ID `EMP001`.

11. To push the code to your GitHub repository, follow the same steps as mentioned in subsection 2.5.3.

Now you have a JavaScript project with examples of maps and classes, and their implementation is verified in the browser's console.


127.0.0.1:5500/index3.html
JavaScript Maps and Classes
Elements Console Sources Network Performance Memory Application
top Filter Default levels 1 Issue: 1
Employee Name: John Doe script3.js:9
Employee ID: EMP001 script3.js:10
City: New York script3.js:11
---------- script3.js:12


127.0.0.1:5500/index3.html
JavaScript Maps and Classes
Elements Console Sources Network Performance Memory Application
top Filter Default levels 1 Issue: 1
Employee Name: Jane Smith script3.js:9
Employee ID: EMP002 script3.js:10
City: San Francisco script3.js:11
---------- script3.js:12


127.0.0.1:5500/index3.html
JavaScript Maps and Classes
Elements Console Sources Network Performance Memory Application
top Filter Default levels 1 Issue: 1
Employee with ID EMP004 not found. script3.js:28