Hands-on Lab: Display Employee Information using Array Methods

Estimated time needed: 30 minutes

What you will learn

In this lab, you will create an employee management system. You will employ buttons to trigger functions such as displaying all employees, calculating total salaries, filtering and displaying HR department employees, and finding employees by their IDs. You will use JavaScript functions to generate dynamic lists of employees utilizimg array methods like forEach, filter, reduce, and find to manage and present data interactively.

Learning objectives

After completing this lab, you will be able to:

- **Event-driven programming:** Learn to trigger functions through button clicks for DOM manipulation.
- **Array method proficiency:** Gain expertise in JavaScript array methods (forEach, filter, reduce, find) for data manipulation.
- **Dynamic manipulation:** Understand how to create and update HTML elements within a webpage dynamically.
- **Front-end development skills:** Develop foundational skills to create interactive interfaces and manage data presentation on webpages.

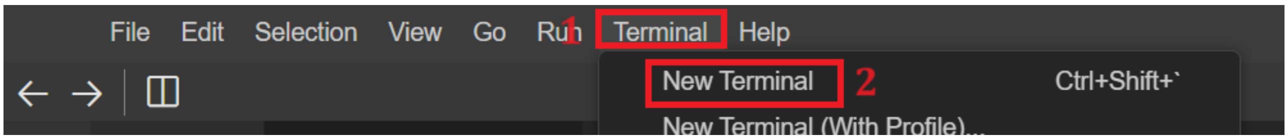
Prerequisites

- Basic Knowledge of HTML and Git commands.
- Basic understanding of JavaScript variables and their scope.
- Web browser with a console (Chrome DevTools, Firefox Console, and so on).

Step 1: Setting up the environment

1. First, you need to clone your main repository in the **Skills Network Environment**. Follow the given steps:

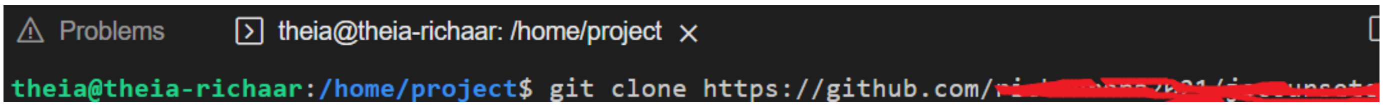
- Click on the terminal in the top-right window pane and then select **New Terminal**.



- Perform `git clone` command by writing given command in the terminal.

```
git clone <github-repository-url>
```

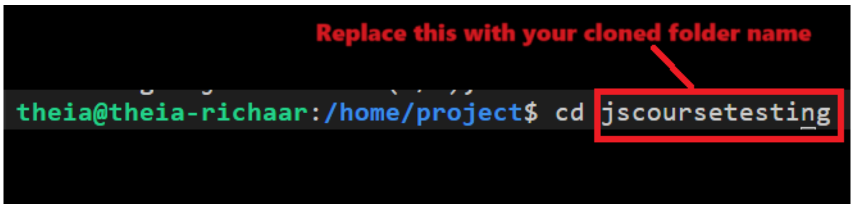
Note: Put your own GitHub repository link instead of `<github-repository-url>` and it should look like as given below:



```
theia@theia-richaar:/home/project$ git clone https://github.com/[redacted]
```

- Above step will clone folder for your GitHub repository under project folder in explorer. You will also see multiple folders inside cloned folder.
- Now you need to navigate inside the cloned folder. For this write given command in the terminal:

```
cd <repository-folder-name>
```



```
theia@theia-richaar:/home/project$ cd jscoursetesting
```

Note: Write your cloned folder name instead of `<repository-folder-name>`. Perform `git clone` if you have logged out of **Skills Network Environment** and you cannot see any files or folder after you logged in.

2. Now, select the **cloned Folder Name** folder, right-click on it, and choose **New Folder**. Enter the folder name as **employeeDetails**. This will create the folder for you. Then, select the **employeeDetails** folder, right-click, and choose **New File**. Enter the file name as **employee_details.html** and click OK. This will create your HTML file.

3. Now select the **employeeDetails** folder again, right click and select **New File**. Enter the file **employee_details.js** and click OK. This process will create your JavaScript file.

4. Create a basic template structure of an HTML file by adding the content provided below.

```
<!DOCTYPE html>
<html>
<head>
  <title>Employee Management System</title>
</head>
<body>
   <h1>Employee Management System</h1>
  <div>
    <button onclick="displayEmployees()">Display Employees</button>
    <button onclick="calculateTotalSalaries()">Calculate Total Salaries</button>
    <button onclick="displayHREmployees()">Display HR Employees</button>
    <button onclick="findEmployeeById(2)">Find Employee by ID 2</button>
  </div>
 <div id="employeesDetails"></div>
<script src="./employee_details.js"></script>
</body>
</html>
```

- It defines buttons within a div to trigger JavaScript functions for managing an employee management system.
- It includes four buttons. Clicking these buttons executes JavaScript functions to display employees, calculate total salaries, filter HR employees, and find an employee by ID.
- HTML code includes three `<div>` to showcase employee information based on user-triggered actions without the page dynamically reloading when clicking on any of the buttons.
- script tag is used in The HTML file above `</body>` tag to include JS file in **employee_details.html**.

Note: After pasting the code, save your file.

Step 2: Defining variables and functions

1. In **employee_details.js**, initialize the employees array object. Initialize it with key value pairs as follows:

```
const employees = [
      { id: 1, name: 'John Doe', age: 30, department: 'IT', salary: 50000 },
      { id: 2, name: 'Alice Smith', age: 28, department: 'HR', salary: 45000 },
      { id: 3, name: 'Bob Johnson', age: 35, department: 'Finance', salary: 60000 },
      //... More employee records can be added here
    ];
```

2. Create the **displayEmployees()** function to display employees details in **employee_details.js** file.

```
// Function to display all employees
function displayEmployees() {
    const totalEmployees = employees
        .map(employee => `<p>${employee.id}: ${employee.name} - ${employee.department} - $${employee.salary}</p>`)
        .join('');
    document.getElementById('employeesDetails').innerHTML = totalEmployees;
}
```

- The map method iterates through each employee in the employees array. For each employee, it constructs a string template literal `<p>${employee.id}: ${employee.name}: ${employee.name} - ${employee.department} - $${employee.salary}</p>`, encapsulated within `<p>` tags, to represent employee details.
- The resulting array of constructed strings is joined into a single string using join(''). This concatenation merges all the HTML-formatted employee details into one continuous string without separators.
- The map method stores employees' details in the variable **totalEmployees**, which shows details in the `<div>` element (with the help of an ID) displays employee information on the webpage.

3. Create the **calculateTotalSalaries()** function to calculate employees' total salaries. Include the codes given below after the previous JavaScript code.

```
function calculateTotalSalaries() {
      const totalSalaries = employees.reduce((acc, employee) => acc + employee.salary, 0);
      alert(`Total Salaries: $${totalSalaries}`);
    }
```

- The reduce method iterates through each employee and accumulates their salaries to calculate the total. The initial value of the accumulator (acc) is 0.
- The reduce method continuously accumulates these salaries by adding each employee's salary to the previous total.
- Each employee's salary (employee.salary) is added to the accumulator (acc). After calculating the total sum of salaries, an alert dialog box is triggered using alert(). It showcases the total calculated salaries with the message "Total Salaries: $" followed by the computed totalSalaries variable value. This alert displays the overall sum of all employee salaries.

4. Create **displayHREmployees()** function to display employees details based on department such as the **HR** department. Include the given code below after the previous JavaScript code.

```
function displayHREmployees() {
     const hrEmployees = employees.filter(employee => employee.department === 'HR');
      const hrEmployeesDisplay = hrEmployees.map((employee, index) => `<p>${employee.id}: ${employee.name}: ${employee.name} - ${employee.departm
      document.getElementById('employeesDetails').innerHTML = hrEmployeesDisplay;
}
```

- The above code filters the employees array using the **filter** array method, isolating and collecting employees whose department property matches 'HR' into a new array named hrEmployees.
- It then displays the matched details on the front page as shown in **displayEmployees** function using map method within `<div>` using its ID **employeesDetails**.

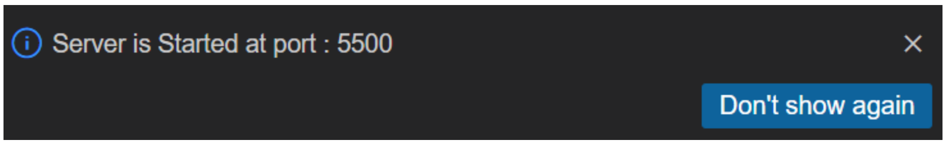
5. Create **findEmployeeById()** function to display employees' details based on ID. Include the below given code after the previous JavaScript code.

```
function findEmployeeById(employeeId) {
      const foundEmployee = employees.find(employee => employee.id === employeeId);
      if (foundEmployee) {
      document.getElementById('employeesDetails').innerHTML =`<p>${foundEmployee.id}: ${foundEmployee.name}: ${foundEmployee.name} - ${foundEmplo
      }
      else{
        document.getElementById('employeesDetails').innerHTML = 'no employee has been found with this ID';
       }
   }
```

- This code uses the find method to locate an employee in the employees array whose ID matches the provided employee ID, storing the found employee object in the foundEmployee variable.
- An if statement is employed to see whether the details for that particular ID are found and display details accordingly.

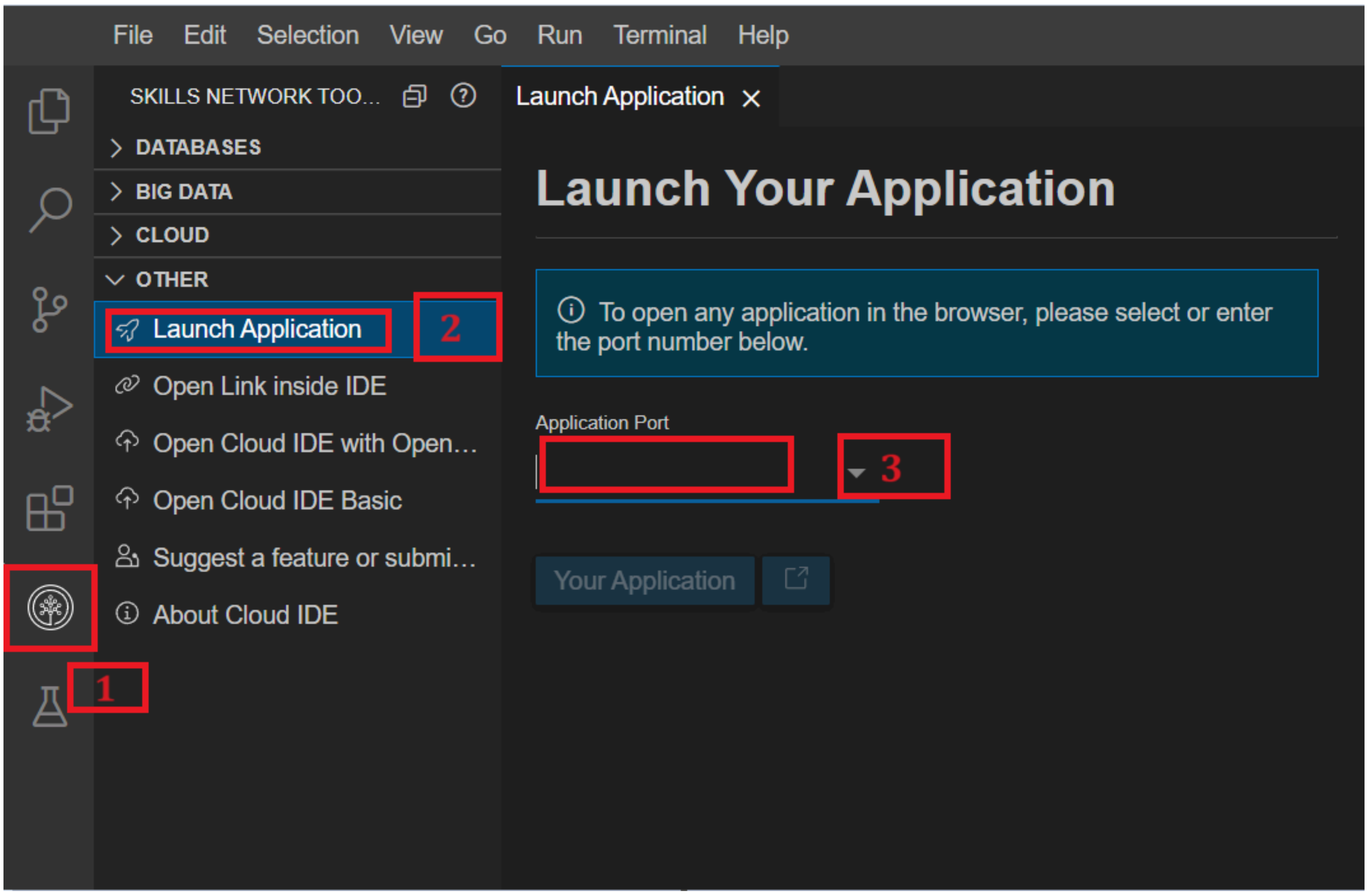
Step 3: Check the output

1. To view your HTML page, right-click the **employee_details.html** file after selecting this file, then select "Open with Live Server".
2. The server should start on port 5500, indicated by a notification on the bottom-right side.



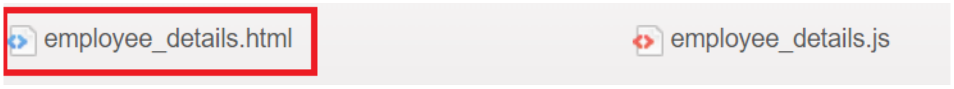
Server is Started at port : 5500

3. Click the Skills Network button on the left (refer to number 1) to open the "Skills Network Toolbox". Next, click **Launch Application** (refer to number 2). From there, you enter port number 5500 at number 3 and click this button.

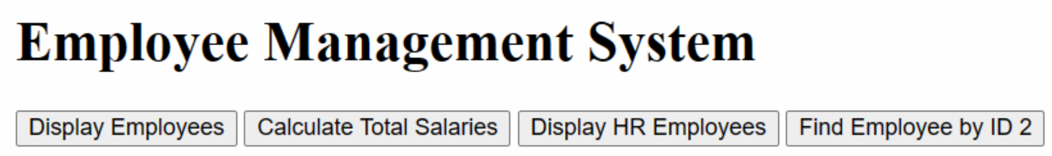


To open any application in the browser, please select or enter the port number below.

4. It will open your default browser, where you will first see the name of your cloned folder. Click on that folder, and inside it, among other folders, you will find the **employeeDetails** folder name. Click on the **employeeDetails** folder, and then select **employee_details.html** file, as shown below.



5. It will open the front page, and you will see the front page as below.



Employee Management System

6. Now, click all the buttons to see the output.

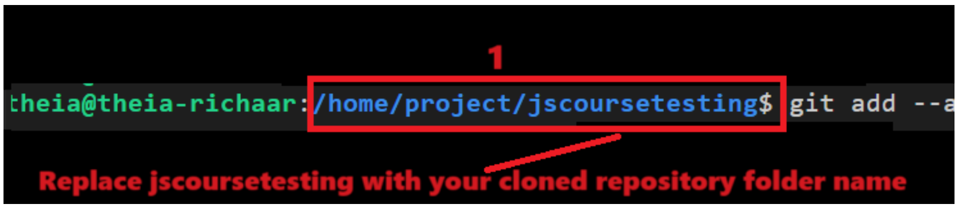
Note: After pasting the code, save your file. You can use any output method for this. If you edit your code, simply refresh your browser running through port number 5500. This way, there is no need to launch the application again and again.

Step 4: Perform Git commands

1. Perform `git add` to add latest files and folder by writing given command in terminal in git environment.

```
git add --a
```

Make sure terminal should have path as follows:



```
theia@theia-richaar:/home/project/jscoursetesting$ git add --a
```

1
Replace jscoursetesting with your cloned repository folder name

2. Then perform `git commit` in the terminal. While performing `git commit`, terminal can show message to set up your `git config --global` for user. name and user.email. If yes, then you need to perform `git config` command as well for `user.name` and `user.email` as given.

   ```
   git config --global user.email "you@example.com"
   ```

   ```
   git config --global user.name "Your Name"
   ```

   Note: Replace data within qoutes with your own details.

   Then perform commit command as given:

   ```
   git commit -m "message"
   ```

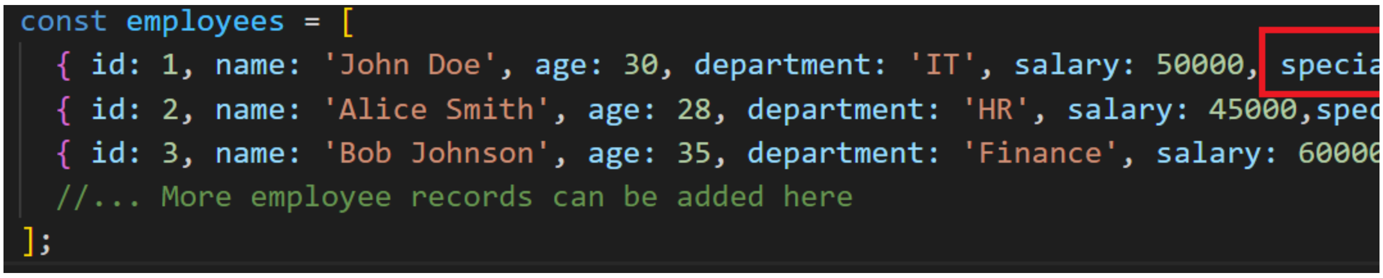
3. Then perform `git push` just by writing given command in terminal.

   ```
   git push origin
   ```

   - After the push command, the system will prompt you to enter your username and password. Enter the username for your GitHub account and the password that you created in the first lab. After entering the credentials, all of your latest folders and files will be pushed to your GitHub repository.

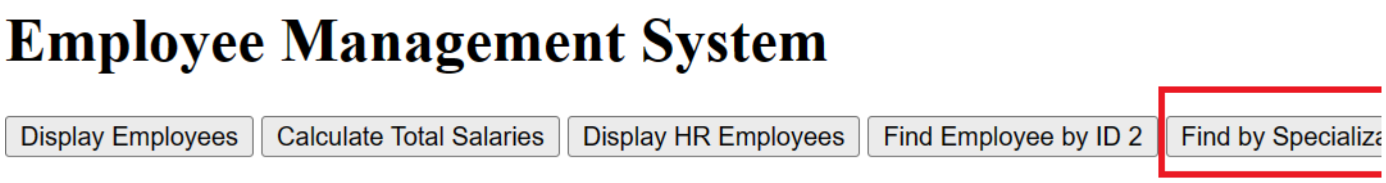
Practice task

1. In this practice task, you need to perform a functionality in which information can be fetched on the basis of employee's speciaization as well.
2. For this, include one more key value pair for employees array of object inside each object as shown in given screenshot.



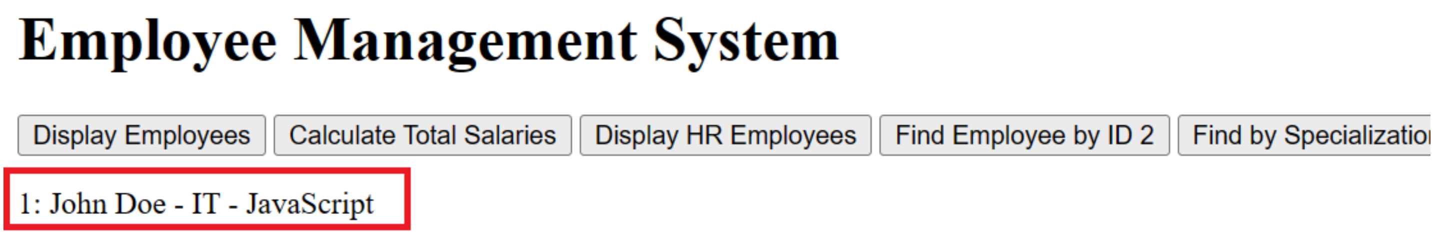
```
const employees = [
  { id: 1, name: 'John Doe', age: 30, department: 'IT', salary: 50000, specia
  { id: 2, name: 'Alice Smith', age: 28, department: 'HR', salary: 45000,spec
  { id: 3, name: 'Bob Johnson', age: 35, department: 'Finance', salary: 6000
  //... More employee records can be added here
];
```

3. Then, create a button to search for an employee based on the specialization as shown in given screenshot.



4. Next, create a JavaScript function to display the details of employees who have specialization in **JavaScript**. You can refer to the **findEmployeeById** function in the JavaScript code of this lab for guidance.

5. The output will be as shown in given screenshot.



1: John Doe - IT - JavaScript

Summary

1. **Setting up the environment:** HTML code includes three `<div>` to showcase employee information based on user-triggered actions without the page dynamically reloading when clicking any buttons.

2. **Defining variables and functions:** The map method stores employees' details in the variable **totalEmployees**, which shows details in the `<div>` element (with the help of an ID) displays employee information on the webpage while the reduce method iterates through each employee and accumulates their salaries to calculate the total.

3. **Checking the output:** `git add`, `git commit`, and `git push` commands update changes into your **Scope_Lab** folder; GitHub repository for proper code management.