**Lab 1**

**Using JSX and ES6 in React**

**By traltb@fe.edu.vn**

Create a React application, using jsx and ES6 in React to complete all exercises 1.

**Exercise 1 – Render employee details using object destructuring**

**Objective**:  
Use ES6 object destructuring to extract properties and render them using JSX.

**Given**:

const employee = { name: "John Doe", age: 30, department: "IT" };

**Task**:  
Render the employee's name inside an <h1>, and their age and department inside <p> tags.

**Exercise 2 – Display a list of employees using map()**

**Objective**:  
Use Array.map() to render a list of employees using <ul> and <li>.

**Given**:

const employees = [

{ id: 1, name: "Anna", department: "HR", age: 50 },

{ id: 2, name: "Brian", department: "IT", age: 40 },

{ id: 3, name: "Clara", department: "Finance", age: 19 },

{ name: "Ann", department: "Finance", age: 22 },

{ name: "Elisabeth", department: "HR", age: 16 }

];

**Task**:  
Render a list showing each employee’s name and department using <ul> and <li>. Use index or id as the key.

**Exercise 3 – Render a table of employees**

**Objective**:  
Use JSX to display employees in a structured table format with headers.

**Task**:  
Render a <table> showing the following columns: ID (or index if missing), Name, Department. Use <thead> and <tbody>.

**Exercise 4 – Calculate average age using rest parameters**

**Objective**:  
Use ES6 rest parameters to collect a dynamic list of ages and calculate their average.

**Task**:  
Create an arrow function averageAge(...ages) that returns the average, and render the result inside a <p>.

**Exercise 5 – Render a dropdown menu of employee names**

**Objective**:  
Use JSX and <select> to create a dropdown from the employee list.

**Task**:  
Map over the employees array and render each name as an <option> in a <select> dropdown.

**Exercise 6 – Filter and display IT department employees**

**Objective**:  
Use Array.filter() and Array.map() to filter employees by department.

**Task**:  
Render only employees from the "IT" department inside a <ul> list.

**Exercise 7 – Sort employees by department, then by name**

**Objective**:  
Sort the employee list first by department alphabetically, then by name if departments match.

**Task**:  
Display the sorted list inside a <ul>. Use Array.sort() and localeCompare() for string sorting.

**Exercise 8 – Group employees by department**

**Objective**:  
Use ES6 and array methods to group employees by department.

**Task**:  
Render department headings (e.g., <h3>HR</h3>) followed by a list of employees in each department using <ul>.

**Exercise 9 – Check if any employee is a teenager**

**Objective**:  
Use Array.some() to check if any employee's age is between 10 and 20.

**Task**:  
Display the result (true or false) inside a <p> tag. You can use:

const isTeenager = employees.some(e => e.age >= 10 && e.age <= 20);

**Exercise 10 – Search for an employee by name**

**Objective**:  
Use Array.find() or filter() to search the array by employee name (case-insensitive).

**Task**:  
Create a simple search bar using <input type="text" />, and display matching employees dynamically as the user types.