NAME: JANHVI SINGH

REG NO: 22BCE8544

MOD 8

HTML CODE:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>LaunchDarkly Employee Dashboard</title>

    <style>

        body {

            font-family: Arial, sans-serif;

            margin: 20px;

            background-color: #f4f4f4;

            color: #333;

        }

        h1 {

            color: #0056b3;

            text-align: center;

            margin-bottom: 30px;

        }

        .container {

            max-width: 1200px;

            margin: 0 auto;

            display: grid;

            grid-template-columns: 1fr 2fr; /\* Form on left, cards on right \*/

            gap: 30px;

        }

        #employee-form-section {

            background-color: #fff;

            padding: 25px;

            border-radius: 8px;

            box-shadow: 0 2px 5px rgba(0,0,0,0.1);

            height: fit-content; /\* Adjust height to content \*/

        }

        #employee-form h2 {

            margin-top: 0;

            color: #0056b3;

        }

        #employee-form label {

            display: block;

            margin-bottom: 5px;

            font-weight: bold;

        }

        #employee-form input[type="text"] {

            width: calc(100% - 20px);

            padding: 10px;

            margin-bottom: 15px;

            border: 1px solid #ddd;

            border-radius: 4px;

        }

        #employee-form button {

            background-color: #007bff;

            color: white;

            border: none;

            padding: 10px 15px;

            border-radius: 5px;

            cursor: pointer;

            font-size: 1em;

            width: 100%;

        }

        #employee-form button:hover {

            background-color: #0056b3;

        }

        #employee-list-section {

            background-color: #fff;

            padding: 25px;

            border-radius: 8px;

            box-shadow: 0 2px 5px rgba(0,0,0,0.1);

        }

        #employee-list-section h2 {

            margin-top: 0;

            color: #0056b3;

        }

        #employee-cards-container {

            display: grid;

            grid-template-columns: repeat(auto-fill, minmax(280px, 1fr));

            gap: 20px;

            margin-top: 20px;

        }

        .employee-card {

            background-color: #f9f9f9;

            border: 1px solid #eee;

            border-left: 5px solid #007bff;

            border-radius: 8px;

            padding: 15px;

            box-shadow: 0 1px 3px rgba(0,0,0,0.1);

            position: relative;

        }

        .employee-card h3 {

            margin-top: 0;

            margin-bottom: 10px;

            color: #333;

        }

        .employee-card p {

            margin: 5px 0;

            font-size: 0.95em;

        }

        .employee-card .actions {

            margin-top: 15px;

            display: flex;

            gap: 10px;

        }

        .employee-card .actions button {

            padding: 8px 12px;

            border: none;

            border-radius: 5px;

            cursor: pointer;

            font-size: 0.85em;

        }

        .employee-card .actions .edit-btn {

            background-color: #ffc107;

            color: #333;

        }

        .employee-card .actions .edit-btn:hover {

            background-color: #e0a800;

        }

        .employee-card .actions .delete-btn {

            background-color: #dc3545;

            color: white;

        }

        .employee-card .actions .delete-btn:hover {

            background-color: #c82333;

        }

    </style>

</head>

<body>

    <h1>Employee Dashboard</h1>

    <div class="container">

        <section id="employee-form-section">

            <form id="employee-form">

                <h2 id="form-title">Add New Employee</h2>

                <input type="hidden" id="employee-id"> <label for="name">Employee Name:</label>

                <input type="text" id="name" required>

                <label for="position">Position:</label>

                <input type="text" id="position" required>

                <label for="department">Department:</label>

                <input type="text" id="department" required>

                <button type="submit" id="submit-btn">Add Employee</button>

            </form>

        </section>

        <section id="employee-list-section">

            <h2>Current Employees</h2>

            <div id="employee-cards-container">

                <p id="no-employees-message">No employees added yet.</p>

            </div>

        </section>

    </div>

    <script src="employees1.js"></script>

</body>

</html>

JS CODE:

// Array to store employee data

let employees = [];

let nextEmployeeId = 1; // Simple counter for unique IDs

let editingEmployeeId = null; // Stores the ID of the employee being edited

// Get DOM elements

const employeeForm = document.getElementById('employee-form');

const formTitle = document.getElementById('form-title');

const employeeIdInput = document.getElementById('employee-id');

const nameInput = document.getElementById('name');

const positionInput = document.getElementById('position');

const departmentInput = document.getElementById('department');

const submitButton = document.getElementById('submit-btn');

const employeeCardsContainer = document.getElementById('employee-cards-container');

const noEmployeesMessage = document.getElementById('no-employees-message');

/\*\*

 \* Renders all employee cards in the DOM.

 \*/

function renderEmployeeCards() {

    employeeCardsContainer.innerHTML = ''; // Clear existing cards

    if (employees.length === 0) {

        noEmployeesMessage.style.display = 'block'; // Show "No employees" message

    } else {

        noEmployeesMessage.style.display = 'none'; // Hide message

        employees.forEach(employee => {

            const employeeCard = document.createElement('div');

            employeeCard.classList.add('employee-card');

            employeeCard.dataset.id = employee.id; // Store ID for easy access

            employeeCard.innerHTML = `

                <h3>${employee.name}</h3>

                <p><strong>Position:</strong> ${employee.position}</p>

                <p><strong>Department:</strong> ${employee.department}</p>

                <div class="actions">

                    <button class="edit-btn" data-id="${employee.id}">Edit</button>

                    <button class="delete-btn" data-id="${employee.id}">Delete</button>

                </div>

            `;

            employeeCardsContainer.appendChild(employeeCard);

        });

    }

    // Attach event listeners to the new buttons

    attachCardButtonListeners();

}

/\*\*

 \* Attaches click listeners to all edit and delete buttons.

 \* This is called after re-rendering to ensure new buttons are interactive.

 \*/

function attachCardButtonListeners() {

    document.querySelectorAll('.edit-btn').forEach(button => {

        button.removeEventListener('click', handleEditEmployee); // Prevent duplicate listeners

        button.addEventListener('click', handleEditEmployee);

    });

    document.querySelectorAll('.delete-btn').forEach(button => {

        button.removeEventListener('click', handleDeleteEmployee); // Prevent duplicate listeners

        button.addEventListener('click', handleDeleteEmployee);

    });

}

/\*\*

 \* Handles form submission for adding or updating an employee.

 \* @param {Event} event - The submit event.

 \*/

function handleFormSubmit(event) {

    event.preventDefault(); // Prevent default form submission and page reload

    const name = nameInput.value.trim();

    const position = positionInput.value.trim();

    const department = departmentInput.value.trim();

    if (!name || !position || !department) {

        alert('All fields are required!');

        return;

    }

    if (editingEmployeeId !== null) {

        // --- Edit Employee Logic ---

        const employeeIndex = employees.findIndex(emp => emp.id === editingEmployeeId);

        if (employeeIndex !== -1) {

            employees[employeeIndex].name = name;

            employees[employeeIndex].position = position;

            employees[employeeIndex].department = department;

        }

        editingEmployeeId = null; // Reset editing state

    } else {

        // --- Add New Employee Logic ---

        const newEmployee = {

            id: nextEmployeeId++,

            name,

            position,

            department

        };

        employees.push(newEmployee);

    }

    employeeForm.reset(); // Clear form fields

    resetFormForAdd(); // Reset form button and title

    renderEmployeeCards(); // Re-render the updated list

}

/\*\*

 \* Handles the click event for the 'Edit' button on an employee card.

 \* Populates the form with the selected employee's data.

 \* @param {Event} event - The click event.

 \*/

function handleEditEmployee(event) {

    const employeeId = parseInt(event.target.dataset.id);

    const employeeToEdit = employees.find(employee => employee.id === employeeId);

    if (employeeToEdit) {

        // Populate form fields

        nameInput.value = employeeToEdit.name;

        positionInput.value = employeeToEdit.position;

        departmentInput.value = employeeToEdit.department;

        employeeIdInput.value = employeeToEdit.id; // Store ID in hidden field

        editingEmployeeId = employeeToEdit.id; // Set editing state

        // Change form title and button text

        formTitle.textContent = 'Edit Employee';

        submitButton.textContent = 'Update Employee';

    }

}

/\*\*

 \* Handles the click event for the 'Delete' button on an employee card.

 \* Removes the employee from the array and updates the DOM.

 \* @param {Event} event - The click event.

 \*/

function handleDeleteEmployee(event) {

    const employeeId = parseInt(event.target.dataset.id);

    const confirmed = confirm(`Are you sure you want to delete this employee (ID: ${employeeId})?`);

    if (confirmed) {

        // Filter out the employee to be deleted

        employees = employees.filter(employee => employee.id !== employeeId);

        // If currently editing the deleted employee, reset the form

        if (editingEmployeeId === employeeId) {

            employeeForm.reset();

            resetFormForAdd();

        }

        renderEmployeeCards(); // Re-render the list

    }

}

/\*\*

 \* Resets the form back to 'Add New Employee' state.

 \*/

function resetFormForAdd() {

    formTitle.textContent = 'Add New Employee';

    submitButton.textContent = 'Add Employee';

    editingEmployeeId = null;

    employeeIdInput.value = ''; // Clear hidden ID

}

// --- Event Listeners ---

employeeForm.addEventListener('submit', handleFormSubmit);

// Initial render when the page loads

renderEmployeeCards();

SCREENSHOT:

