Leave Management System

Phase - 1
* let's keep simple logics in order to understand complex technical stuffs
* we going to use (fetch api) for connecting frontend app build with [HTML , CSS , JavaScript]
* we going to use (async , await) to keep a readable code
* we going to use json-server as data provider for Rest API

Task :

Employees are working in an organization called UnicomTIC , they entitled to apply leaves. once the leave request applied by employee , manager level staffs can view the leave request and they can approve it , once it done final approval will be make by director/CEO of organization.

To keep things simple we will stick with this. In future we will add leave Types, leave allocations and more complex logic

In more detail the responsibility of users are as follows

Admin:
* can create,edit,delete,view,search all employees

Employee
* view his/her employee information
* Apply his/her Leave [Goes to Manager and then CEO]
* View his/her Status of Leave Requested
* View his/her Leave History

Manager
* View All Employees
* Apply his/her Leave [Goes to CEO]
* View his/her Status of Leave Requested
* View All leaves Applied by other employee
* Accept or Reject Leave

ceo
* View All Employees
* Apply his/her Leave [Goes to CEO]
* View his/her Status of Leave Requested
* View All leaves Applied by other employee
* Accept or Reject Leave

For this system you required to the following activities step by step
1. Let's perform CRUD operations with search for Employee once it done credential also have to created

Employee	: {EMP ID , NIC number , first Name , Last Name , DOB , DOJ }
users	: {EMP ID , username , password , role}

2. Going to create Leave Application forms and View the Appiled leaves
3. Let's perform login activities and check whether it navigates to approriate Page , with appropriate Menus
    [Going to use Single Page App Mechanism , instead of creating 4 pages]
4. Further refinement on Leave processing Flow

Steps:

Create Folder —->
Backend
FrontEnd

Command

Npm init -y
Npm install json-server

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Employee Management</title>
    <link rel="stylesheet" href="styles.css">
</head>

<body>

    <div class="container">
        <div class="flex-container">
            <h1>Employee Management</h1>
            <!-- Add Employee Button -->
            <button id="createEmployeeBtn">Add Employee</button>
        </div>

        <!-- Search Input -->
        <input type="text" id="searchInput" placeholder="Search by NIC, First Name, Last
Name..."
            style="margin-top: 10px;">
```

```html
<!-- Employee Table -->


<table id="employeeTable">
    <thead>
        <tr>
            <th>EMP ID</th>
            <th>NIC</th>
            <th>First Name</th>
            <th>Last Name</th>
            <th>Date of Birth</th>
            <th>Date of Joining</th>
            <th>Actions</th>
        </tr>
    </thead>
    <tbody id="employeeTableBody"></tbody>
</table>

<br>

<!-- Toast Notification -->
<div id="toast"></div>

<!-- Employee Modal -->
<div id="employeeModal" class="modal">
    <div class="modal-content">
        <span class="close">&times;</span>
        <h2 id="modalTitle">Add Employee</h2>
        <form id="employeeForm">
            <input type="hidden" id="empId" name="empId">

            <div>
                <label for="nic">NIC Number:</label>
                <input type="text" id="nic" name="nic" required>
            </div>

            <div>
                <label for="firstName">First Name:</label>
                <input type="text" id="firstName" name="firstName" required>
            </div>

            <div>
                <label for="lastName">Last Name:</label>
                <input type="text" id="lastName" name="lastName" required>
            </div>

            <div>
                <label for="dob">Date of Birth:</label>
                <input type="date" id="dob" name="dob" required>
            </div>

            <div>
                <label for="doj">Date of Joining:</label>
                <input type="date" id="doj" name="doj" required>
            </div>
            <button type="submit" id="saveButton">Save</button>
        </form>
    </div>
</div>

<!-- Credentials Modal -->
<div id="credentialsModal" class="modal">
    <div class="modal-content">
        <span class="close">&times;</span>
        <h2>Edit Credentials</h2>
        <form id="credentialsForm">
            <input type="hidden" id="credentialsEmpId" name="empId">

            <div>
                <label for="username">Username:</label>
                <input type="text" id="username" name="username" disabled required>
            </div>

            <div>
                <label for="password">Password:</label>
                <input type="text" id="password" name="password" required>
            </div>

            <div>
```

```html
                        <label for="userrole">Role:</label>
                        <select id="userrole" name="userrole">
                            <option value="Employee">Employee</option>
                            <option value="Manager">Manager</option>
                            <option value="Admin">Admin</option>
                            <option value="Director">Director</option>
                        </select>
                    </div>
                    <button type="submit">Save Credentials</button>
                </form>
            </div>
        </div>





    </div>
    <script src="./script.js"></script>
</body>

</html>
```

```javascript
document.addEventListener("DOMContentLoaded", async () => {
    const apiUrl = "http://localhost:3000/employees";
    const userApiUrl = "http://localhost:3000/users";

    let employees = [];
    let users = [];
    let editingEmployeeId = null;

    const employeeTableBody = document.getElementById("employeeTableBody");
    const employeeModal = document.getElementById("employeeModal");
    const credentialsModal = document.getElementById("credentialsModal");
    const employeeForm = document.forms['employeeForm'];
    const credentialsForm = document.forms['credentialsForm'];
    const toast = document.getElementById("toast");

    // Fetch Employees
    const fetchEmployees = async () => {
        const response = await fetch(apiUrl);
        employees = await response.json();
        renderEmployees();
    };

    // Fetch Users
    const fetchUsers = async () => {
        const response = await fetch(userApiUrl);
        users = await response.json();
    };

    // Render Employees
    const renderEmployees = () => {
        employeeTableBody.innerHTML = "";
        employees.forEach((employee) => {
            const row = document.createElement("tr");
            row.innerHTML = `
                <td>${employee.empId}</td>
                <td>${employee.nic}</td>
                <td>${employee.firstName}</td>
                <td>${employee.lastName}</td>
                <td>${employee.dob}</td>
                <td>${employee.doj}</td>
                <td>
                    <button onclick="editEmployee('${employee.empId}')">Edit</button>
                    <button onclick="deleteEmployee('${employee.empId}')">Delete</button>
                    <button onclick="editCredentials('${employee.empId}')">Edit
Credentials</button>
                </td>
            `;
            employeeTableBody.appendChild(row);
        });
    };

    // Open Employee Modal for Create or Edit
    document.getElementById("createEmployeeBtn").onclick = () => {
        openEmployeeModal();
    };

    const openEmployeeModal = (employee = null) => {
        employeeModal.style.display = "block";
        const modalTitle = document.getElementById("modalTitle");
        const saveButton = document.getElementById("saveButton");

        if (employee) {
            modalTitle.innerText = "Edit Employee";
            saveButton.innerText = "Update";
            employeeForm.nic.value = employee.nic;
            employeeForm.firstName.value = employee.firstName;
            employeeForm.lastName.value = employee.lastName;
            employeeForm.dob.value = employee.dob;
            employeeForm.doj.value = employee.doj;
            editingEmployeeId = employee.empId;
        } else {
            employeeForm.reset();
            modalTitle.innerText = "Add Employee";
            saveButton.innerText = "Save";
            editingEmployeeId = null;
        }
    };

    // Close Modal
    document.querySelectorAll(".close").forEach(btn => {
        btn.onclick = () => {
            employeeModal.style.display = "none";
            credentialsModal.style.display = "none";
        };
    });

    // Function to generate unique EMP ID
    const generateEmpId = () => {
        const characters =
'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789';
        let empId;
        do {
            empId = 'EMP_' + Array.from({ length: 5 }, () =>
characters.charAt(Math.floor(Math.random() * characters.length))).join('');
        } while (employees.some(emp => emp.empId === empId));
        return empId;
    };
```

```javascript
    // Save Employee
    employeeForm.onsubmit = async (e) => {
        e.preventDefault();

        const empId = editingEmployeeId || generateEmpId();
        const nic = employeeForm.nic.value;
        const firstName = employeeForm.firstName.value;
        const lastName = employeeForm.lastName.value;
        const dob = employeeForm.dob.value;
        const doj = employeeForm.doj.value;

        const employeeData = {
            id: empId,
            empId: empId,
            nic: nic,
            firstName: firstName,
            lastName: lastName,
            dob: dob,
            doj: doj
        };

        if (editingEmployeeId) {
            // Update employee
            await fetch(`${apiUrl}/${editingEmployeeId}`, {
                method: "PUT",
                headers: {
                    "Content-Type": "application/json"
                },
                body: JSON.stringify(employeeData)
            });
        } else {
            // Create new employee
            await fetch(apiUrl, {
                method: "POST",
                headers: {
                    "Content-Type": "application/json"
                },
                body: JSON.stringify(employeeData)
            });
            // Automatically create user credentials
            const userData = {
                id: empId,
                empId: empId,
                username: empId,
                password: "pwd123",
                role: "Employee"  // Default role as Employee; can be modified as needed
            };
            await fetch(userApiUrl, {
                method: "POST",
                headers: {
                    "Content-Type": "application/json"
                },
                body: JSON.stringify(userData)
            });
        }

        employeeModal.style.display = "none";
        await fetchEmployees();
        await fetchUsers(); // To refresh user data if needed
    };

    // Delete Employee
    window.deleteEmployee = async (empId) => {
        if (confirm("Are you sure you want to delete this employee?")) {
            await fetch(`${apiUrl}/${empId}`, {
                method: "DELETE"
            });
            await fetch(`${userApiUrl}/${empId}`, {
                method: "DELETE"
            });
            showToast("Deleted Successfully");
            await fetchEmployees();
        }
    };

    // Edit Employee
    window.editEmployee = (empId) => {
        const employee = employees.find(emp => emp.empId === empId);
        openEmployeeModal(employee);
    };

    // Open Credentials Modal
    window.editCredentials = (empId) => {
        const user = users.find(usr => usr.empId === empId);
        if (user) {
            credentialsModal.style.display = "block";
            credentialsForm.username.value = user.username;
            credentialsForm.password.value = user.password;

            credentialsForm.userrole.value = user.role;
            credentialsForm.credentialsEmpId.value = user.empId;
        }
    };

    // Save Credentials
    credentialsForm.onsubmit = async (e) => {
        e.preventDefault();
        const userData = {
            username: credentialsForm.username.value,
            password: credentialsForm.password.value,
            role: credentialsForm.userrole.value,
            empId: credentialsForm.credentialsEmpId.value
        };
        const user = users.find(usr => usr.empId === userData.empId);
        if (user) {
            await fetch(`${userApiUrl}/${user.id}`, {
                method: "PUT",
                headers: {
                    "Content-Type": "application/json"
                },
                body: JSON.stringify(userData)
            });
        } else {
            await fetch(userApiUrl, {
                method: "POST",
                headers: {
                    "Content-Type": "application/json"
                },
                body: JSON.stringify(userData)
            });
        }
        credentialsModal.style.display = "none";
        await fetchUsers();
    };

    // Show Toast
    const showToast = (message) => {
        toast.innerText = message;
        toast.classList.add("show");
        setTimeout(() => {
            toast.classList.remove("show");
        }, 3000);
    };

    // Search Employees
    document.getElementById("searchInput").oninput = (e) => {
        const query = e.target.value.toLowerCase();
        const filteredEmployees = employees.filter(emp => {
            return emp.nic.toLowerCase().includes(query) ||
                emp.firstName.toLowerCase().includes(query) ||
                emp.lastName.toLowerCase().includes(query);
        });
        renderFilteredEmployees(filteredEmployees);
    };

    const renderFilteredEmployees = (filteredEmployees) => {
        employeeTableBody.innerHTML = "";
        filteredEmployees.forEach((employee) => {
            const row = document.createElement("tr");
            row.innerHTML = `
                <td>${employee.empId}</td>
                <td>${employee.nic}</td>
                <td>${employee.firstName}</td>
                <td>${employee.lastName}</td>
                <td>${employee.dob}</td>
                <td>${employee.doj}</td>
                <td>
                    <button onclick="editEmployee('${employee.empId}')">Edit</button>
                    <button onclick="deleteEmployee('${employee.empId}')">Delete</button>
                    <button onclick="editCredentials('${employee.empId}')">Edit
Credentials</button>
                </td>
            `;
            employeeTableBody.appendChild(row);
        });
    };

    // Initialize
    fetchEmployees();
    fetchUsers();
});
```

```
*{
    margin: 0;
    padding: 0;
}

body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 20px;
    background-color: #f4f4f4;
}

.container {
    max-width: 1000px;
    margin: 0 auto;
    background: #fff;
    padding: 20px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}

.flex-container{
    display: flex;
    flex-direction: row;
    justify-content: space-between;
}
```

```css
h1 {
    text-align: center;
}

#searchInput {
    width: 100%;
    padding: 10px;
    margin-bottom: 20px;
    box-sizing: border-box;
}

#employeeTable {
    width: 100%;
    border-collapse: collapse;
    margin-bottom: 20px;
}

#employeeTable th, #employeeTable td {
    border: 1px solid #ddd;
    padding: 8px;
}

#employeeTable th {
    background-color: #f2f2f2;
    text-align: left;
}

#employeeTable tr:nth-child(even) {
    background-color: #f9f9f9;
}

button {
    padding: 10px 20px;
    background-color: #4CAF50;
    color: white;
    border: none;
    cursor: pointer;
}

button:hover {
    background-color: #45a049;
}

.modal {
    display: none;
    position: fixed;
    z-index: 1;
    padding-top: 100px;
    left: 0;
    top: 0;
    width: 100%;
    height: 100%;
    background-color: rgba(0, 0, 0, 0.4);
}

.modal-content {
    background-color: white;
    margin: auto;
    padding: 20px;
    border: 1px solid #888;
    width: 50%;
    box-sizing: border-box;
}

.close {
    color: #aaa;
    float: right;
    font-size: 28px;
    font-weight: bold;
}

.close:hover, .close:focus {
    color: black;
    text-decoration: none;
    cursor: pointer;
}

.toast {
    visibility: hidden;
    min-width: 250px;
    margin-left: -125px;
    background-color: #333;
    color: #fff;
    text-align: center;
    border-radius: 2px;
    padding: 16px;
    position: fixed;
    z-index: 1;
    left: 50%;
    bottom: 30px;
    font-size: 17px;
}

.toast.show {
    visibility: visible;
    animation: fadein 0.5s, fadeout 0.5s 2.5s;
}

@keyframes fadein {
    from {bottom: 0; opacity: 0;}
    to {bottom: 30px; opacity: 1;}
}

@keyframes fadeout {
    from {bottom: 30px; opacity: 1;}
    to {bottom: 0; opacity: 0;}
}
```

```json
{
  "name": "leave_management",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "json-server --watch db.json --port 3000"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": "",
  "dependencies": {
    "json-server": "^1.0.0-beta.2"
  }
}
```

Db.json

```
{
  "employees": [
    {
      "id": "EMP_KaYVb",
      "empId": "EMP_KaYVb",
      "nic": "12345V",
      "firstName": "Raju",
      "lastName": "Bai",
      "dob": "2024-09-02",
      "doj": "2024-09-01"
    },
    {
      "id": "EMP_Ka001",
      "empId": "EMP_Ka001",
      "nic": "12345V",
      "firstName": "Raju",
      "lastName": "Bai",
      "dob": "2024-09-02",
      "doj": "2024-09-01"
    }
  ],
  "users": [
    {
      "id": "EMP_KaYVb",
      "empId": "EMP_KaYVb",
      "username": "EMP_KaYVb",
      "password": "pwd123",
      "role": "Employee"
    }
  ],
  "leaveRequests": [
    {
      "id": 1,
      "employeeId": "EMP_KaYVb",
      "reason":"sick",
      "dateFrom":"2024-09-12",
      "numOfDays":2
    },
    {
      "id": 2,
      "employeeId": "EMP_KaYVb",
      "reason":"pickic",
      "dateFrom":"2024-09-12",
      "numOfDays":4
    },
    {
      "id": 3,
      "employeeId": "EMP_Ka001",
      "reason":"pickic",
      "dateFrom":"2024-09-12",
      "numOfDays":4
    }
  ]
}
```