**Q4: Efficient Storage using Hash Tables**: Design a **hash table** to store and retrieve employee records based on employee IDs. Implement different **hash functions** and collision handling techniques (chaining, open addressing).

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Hash Table - Employee Records</title>

<style>

body { font-family: Arial, sans-serif; padding: 20px; }

.container { max-width: 600px; margin: 0 auto; }

input, button { padding: 10px; margin: 5px; }

table { width: 100%; border-collapse: collapse; margin-top: 20px; }

th, td { padding: 10px; border: 1px solid #ddd; text-align: left; }

</style>

</head>

<body>

<div class="container">

<h1>Employee Records - Hash Table</h1>

<label for="empId">Employee ID:</label>

<input type="text" id="empId" placeholder="Enter Employee ID" />

<label for="empName">Employee Name:</label>

<input type="text" id="empName" placeholder="Enter Employee Name" />

<button onclick="addRecord()">Add Record</button>

<h2>Employee Records</h2>

<table id="employeeTable">

<thead>

<tr>

<th>Employee ID</th>

<th>Name</th>

</tr>

</thead>

<tbody>

<!-- Records will be inserted here -->

</tbody>

</table>

</div>

<script>

class HashTable {

constructor(size) {

this.size = size;

this.table = new Array(size);

}

// Simple hash function

hash(key) {

return key % this.size;

}

add(key, value) {

const index = this.hash(key);

if (!this.table[index]) {

this.table[index] = [];

}

this.table[index].push([key, value]); // Chaining

}

get(key) {

const index = this.hash(key);

const bucket = this.table[index];

if (bucket) {

for (let i = 0; i < bucket.length; i++) {

if (bucket[i][0] === key) {

return bucket[i][1];

}

}

}

return null; // Not found

}

}

const hashTable = new HashTable(10);

function addRecord() {

const empId = parseInt(document.getElementById('empId').value);

const empName = document.getElementById('empName').value;

if (empId && empName) {

hashTable.add(empId, empName);

updateTable();

}

}

function updateTable() {

const tbody = document.getElementById('employeeTable').getElementsByTagName('tbody')[0];

tbody.innerHTML = '';

for (let i = 0; i < hashTable.size; i++) {

const bucket = hashTable.table[i];

if (bucket) {

for (let [empId, empName] of bucket) {

const row = tbody.insertRow();

row.insertCell(0).innerText = empId;

row.insertCell(1).innerText = empName;

}

}

}

}

</script>

</body>

</html>