**Real-time Seating Arrangement:**

**Create a 2-D array to represent seating arrangements for a movie theatre. The system should handle seat booking and cancellation requests dynamically. Add constraints for group bookings, ensuring they are seated together.**

<!DOCTYPE html>

<html>

<head>

<title>Movie Theater Seating</title>

<style>

body {

font-family: Arial, sans-serif;

}

table {

margin: 20px 0;

border-collapse: collapse;

}

th, td {

border: 1px solid #000;

padding: 10px;

text-align: center;

}

.booked {

background-color: #ffcccc;

}

input {

margin: 5px;

}

</style>

</head>

<body>

<h1>Movie Theater Seating Arrangement</h1>

<h2>Seating Chart</h2>

<table id="seatingChart"></table>

<h2>Book Seats</h2>

<form id="bookingForm">

<input type="number" id="groupSize" placeholder="Group Size" min="1" required>

<button type="submit">Book Seats</button>

</form>

<h2>Cancel Booking</h2>

<form id="cancellationForm">

<input type="number" id="seatRow" placeholder="Row (0-indexed)" required>

<input type="number" id="seatCol" placeholder="Column (0-indexed)" required>

<button type="submit">Cancel Booking</button>

</form>

<script>

const rows = 5; // Number of rows

const cols = 5; // Number of columns

const seatingArrangement = [];

// Create the 2D array manually

for (let i = 0; i < rows; i++) {

seatingArrangement[i] = []; // Initialize each row

for (let j = 0; j < cols; j++) {

seatingArrangement[i][j] = false; // Fill with false

}

}

// Function to render the seating chart

function renderSeatingChart() {

const seatingChart = document.getElementById('seatingChart');

seatingChart.innerHTML = ''; // Clear existing chart

for (let i = 0; i < rows; i++) {

const row = document.createElement('tr');

for (let j = 0; j < cols; j++) {

const seat = document.createElement('td');

seat.textContent = (i) + '-' + (j); // Display seat number

seat.className = seatingArrangement[i][j] ? 'booked' : '';

seat.onclick = () => toggleSeat(i, j);

row.appendChild(seat);

}

seatingChart.appendChild(row);

}

}

// Function to book seats

function bookSeats(groupSize) {

for (let i = 0; i < rows; i++) {

for (let j = 0; j <= cols - groupSize; j++) {

// Check if seats are available

let canBook = true;

for (let k = 0; k < groupSize; k++) {

if (seatingArrangement[i][j + k]) {

canBook = false;

break;

}

}

// If available, book the seats

if (canBook) {

for (let k = 0; k < groupSize; k++) {

seatingArrangement[i][j + k] = true; // Mark seats as booked

}

renderSeatingChart();

alert('Seats booked successfully!');

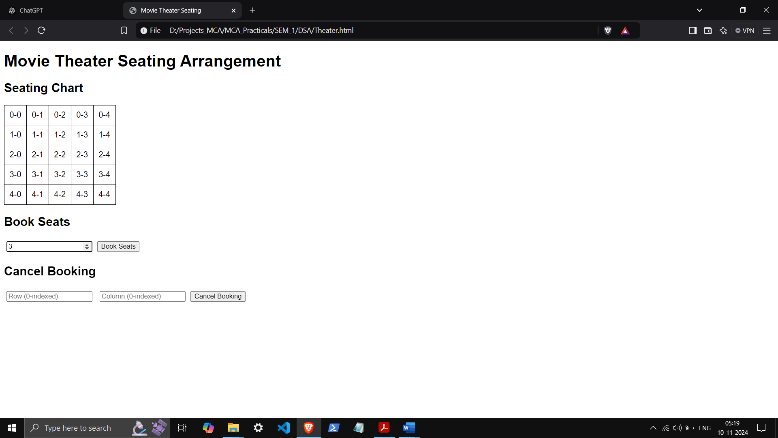
return;

}

}

}

alert('Not enough contiguous seats available for this group!');

 }

// Function to cancel a booking

function cancelBooking(row, col) {

if (seatingArrangement[row][col]) {

seatingArrangement[row][col] = false; // Mark seat as available

renderSeatingChart();

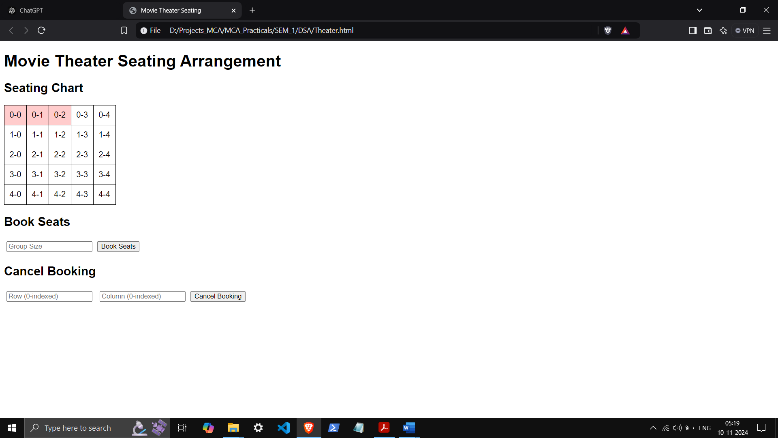
alert('Booking canceled successfully!');

} else {

alert('No booking found for this seat!');

}

}



// Function to toggle seat selection for booking

function toggleSeat(row, col) {

if (seatingArrangement[row][col]) {

cancelBooking(row, col);

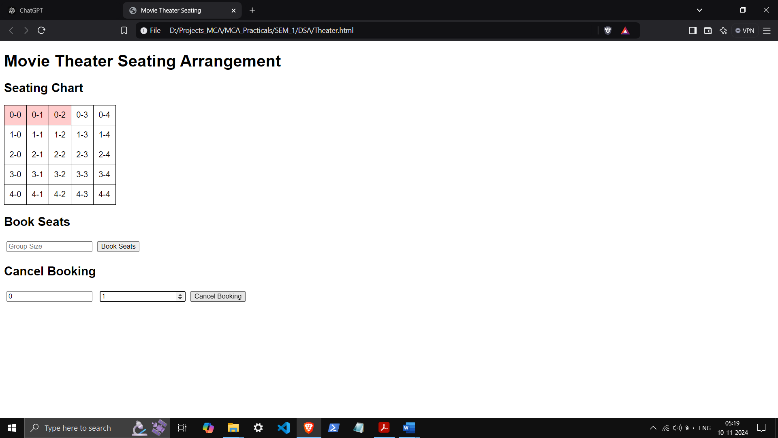
}

}

// Event listeners for forms

document.getElementById('bookingForm').addEventListener('submit', function(event) {

event.preventDefault();

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

bookSeats(groupSize);

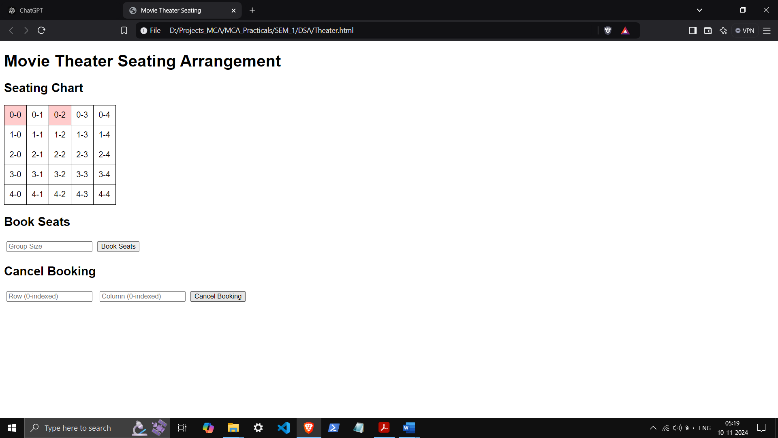
this.reset();

});

document.getElementById('cancellationForm').addEventListener('submit', function(event) {

event.preventDefault();

const row = parseInt(document.getElementById('seatRow').value);

 const col = parseInt(document.getElementById('seatCol').value);

cancelBooking(row, col);

this.reset();

});

// Initial render of the seating chart

renderSeatingChart();

</script>

</body>

</html>