SUMMER INTERNSHIP

on

Full-Stack Web Development

Submitted by

Velpula Mahendra

Registration No.: 12305538

Program Name: MCA

School of Computer Application

Lovely Professional University, Phagwara

(June – July 2024)

Acknowledgment

I want to express my heartfelt thanks to Udemy and Dr. Angela Yu for the incredible learning experience I had during my internship. I'm truly grateful for the opportunity to learn from Dr. Angela Yu, whose teaching style made complex topics easy to understand and enjoyable. Her passion for the subject really inspired me and kept me motivated throughout the course.

I also want to acknowledge the effort and dedication of everyone at Udemy for creating such a supportive learning environment. The resources and guidance provided were invaluable, and I feel lucky to have had the chance to be part of this amazing journey. Thank you for making this experience so rewarding and memorable!

Velpula Mahendra

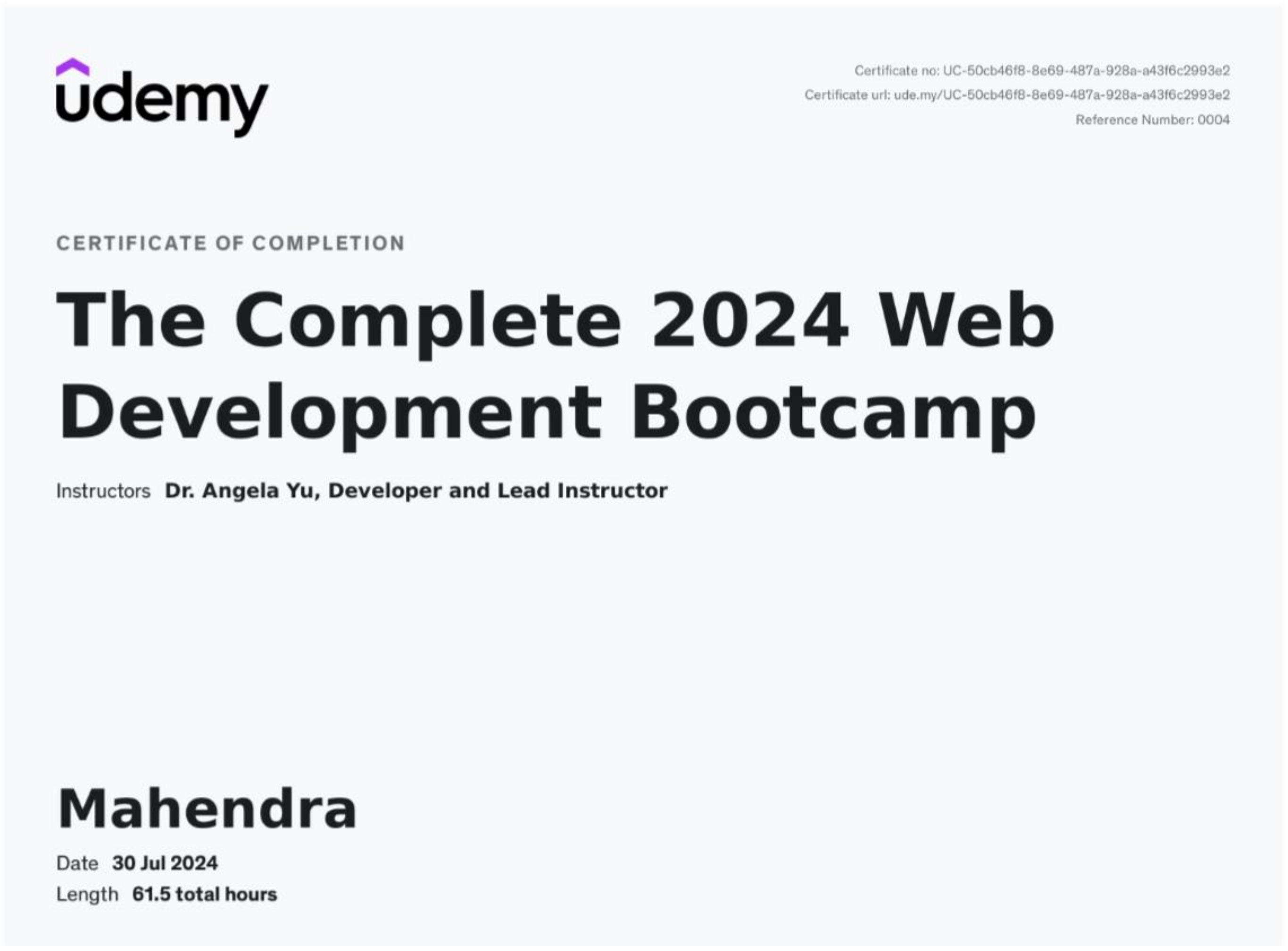
Registration no: 12305538

Project Github Link: **https://github.com/Mahendra-ghub/web_technologies/tree/master/Movieticketbooking**

Certificate

Link: https://www.udemy.com/certificate/UC-50cb46f8-8e69-487a-928a-a43f6c2993e2/

https://www.udemy.com/certificate/UC-50cb46f8-8e69-487a-928a-a43f6c2993e2/



udemy

Certificate no: UC-50cb46f8-8e69-487a-928a-a43f6c2993e2
Certificate url: ude.my/UC-50cb46f8-8e69-487a-928a-a43f6c2993e2
Reference Number: 0004

CERTIFICATE OF COMPLETION

The Complete 2024 Web Development Bootcamp

Instructors **Dr. Angela Yu, Developer and Lead Instructor**

Mahendra

Date **30 Jul 2024**
Length **61.5 total hours**

Table of contents

1.Introduction of the Course:

Syllabus Breakdown

1. Web Development Fundamentals

- **Overview:** Understand how websites function and the role of HTML, CSS, and JavaScript.
- **Topics Covered:**
    - Basics of web technology and website serving
    - Introduction to HTML, CSS, and JavaScript

2. HTML Fundamentals

- **Overview:** Learn the core structure and syntax of HTML to create and organize web content.
- **Topics Covered:**
    - Anatomy of HTML syntax
    - HTML boilerplate and doctypes
    - Indentation and nesting practices
    - HTML tags for headings, paragraphs, and lists
    - Inserting images and creating hyperlinks
    - Developing multi-page websites
    - HTML best practices

3. CSS Fundamentals

- **Overview:** Master CSS to style and layout web pages effectively.
- **Topics Covered:**
    - Basics of CSS and its role in styling
    - Inline, internal, and external CSS
    - CSS selectors and properties
    - Specificity and inheritance
    - The CSS Box Model
    - Positioning and display properties
    - Responsive design with media queries
    - CSS float, clear properties, and selector combinations

4. Flexbox

- **Overview:** Utilize Flexbox for flexible and efficient layout design.
- **Topics Covered:**
    - Introduction to Flexbox and its benefits
    - Flex direction and alignment
    - Flex shorthand property
    - Sizing and distributing child items within a flex container

5. Bootstrap

- **Overview:** Use Bootstrap framework to quickly build responsive and visually appealing websites.
- **Topics Covered:**
    - Basics of Bootstrap and its 12-column grid system
    - Incorporating Bootstrap components (buttons, carousels, cards, navigation bars)
    - Using Bootstrap design elements and pre-built templates

6. Document Object Model (DOM)

- **Overview:** Manipulate and interact with HTML elements using JavaScript.
- **Topics Covered:**
    - Tree structure of HTML documents
    - Navigation and manipulation using object notation
    - Best practices for separation of concerns and code cleanliness

7. Grid Layout

- **Overview:** Create complex layouts using CSS Grid Layout.
- **Topics Covered:**
    - Differences between Grid and Flexbox
    - Setting up a grid and positioning items
    - Combining Grid and Flexbox for advanced layouts

8. Web Design Principles

- **Overview:** Apply design principles to create user-friendly and visually appealing websites.
- **Topics Covered:**

    - Fundamental principles of UI and UX design
    - Color theory and modern typography
    - Managing user attention and creating a professional look

9. JavaScript ES6

- **Overview:** Dive into JavaScript to add interactivity and functionality to websites.
- **Topics Covered:**
    - Basic syntax and data types (variables, operators, control structures)
    - Functions (declarations, expressions, arrow functions)
    - Arrays and objects (methods, iteration)
    - Object-oriented programming concepts
    - Advanced features (spread syntax, higher-order functions)

10. Git and Version Control

- **Overview:** Learn to manage code changes and collaborate using Git and GitHub.
- **Topics Covered:**
    - Basics of Git for version control
    - Key concepts: forking, branching, cloning
    - Using GitHub as a remote repository for collaboration

Brief Description of the Work Done

Position of Internship and Roles

Project Overview: The movie ticket booking project aimed to develop a comprehensive web application that allows users to browse movie listings, select showtimes, and purchase tickets. The application was built using HTML, CSS, and JavaScript to ensure a seamless, responsive, and interactive user experience. The project involved both frontend and backend development, including database management and user interface design.

Roles and Responsibilities:

1. **Frontend Development:**
    - **HTML/CSS:**
        - Created the structure of the application using HTML, ensuring that the layout was well-organized and easy to navigate.
        - Applied CSS to style the application, focusing on creating a visually appealing design that was also responsive across various devices (desktops, tablets, and mobile phones). Utilized CSS Grid and Flexbox to handle responsive design and ensure a consistent look and feel.
    - **JavaScript:**
        - Developed interactive features such as dropdown menus for selecting movie showtimes, dynamic updating of available tickets, and client-side validation of booking forms.
        - Implemented client-side logic to handle user inputs and interactions, including calculating ticket prices, managing seat availability, and updating the UI based on user actions.
2. **Backend Development:**
    - **Server-Side Logic:**
        - Wrote server-side scripts to handle various aspects of the booking process, such as processing user requests, managing seat reservations, and updating ticket availability.
        - Implemented business logic to ensure that bookings were processed correctly and that no double bookings occurred.
    - **Database Integration:**

        - Integrated a backend database (e.g., MongoDB) to store information about movies, showtimes, and user bookings. Designed the database schema to efficiently handle queries and updates related to ticket sales and availability.

3. **Database Management:**
    - **Database Design:**
        - Designed and implemented the database schema, which included tables for movies, showtimes, users, and bookings. Ensured that the schema supported efficient querying and data retrieval.
        - Created indexes on frequently queried fields to improve performance and reduce query execution time.
    - **Data Operations:**
        - Implemented CRUD (Create, Read, Update, Delete) operations to manage movie listings, showtimes, and booking records. Ensured that data integrity was maintained throughout the application.
4. **User Interface Design:**
    - **UI/UX Design:**
        - Focused on creating a user-friendly interface that made it easy for users to browse movies, select showtimes, and book tickets. Emphasized usability and accessibility to ensure that the application was intuitive and easy to use.
        - Conducted user testing sessions to gather feedback on the UI/UX design and made iterative improvements based on user input.
5. **Testing and Debugging:**
    - **Functionality Testing:**
        - Performed comprehensive testing to ensure that all features of the application, such as movie listings, showtime selection, and booking processes, worked as expected.
        - Used testing frameworks and tools (e.g., Jest for unit tests) to validate that individual components and overall functionality were error-free.
    - **Debugging:**
        - Identified and resolved bugs related to frontend and backend components. Utilized debugging tools and techniques to track down and fix issues, including inspecting browser console logs and server logs.
6. **Deployment:**

- **Hosting:**
    - Deployed the application to a web server (e.g., Heroku or AWS) to make it accessible to users. Configured server settings and ensured that the application was live and functioning as intended.
- **Maintenance:**
    - Monitored the application post-deployment to identify and address any issues that arose. Implemented updates and improvements based on user feedback and performance metrics.

Challenges Faced and Solutions:

1. **Handling High Traffic:**
    - **Challenge:** The application needed to manage high traffic during peak booking times, which could lead to performance issues and slow response times.
    - **Solution:** Implemented code splitting and lazy loading techniques to optimize the application's performance. Used server-side caching and database indexing to handle large volumes of data efficiently.
2. **Ensuring Data Accuracy:**
    - **Challenge:** Keeping booking information accurate and up-to-date in real-time was crucial to prevent double bookings and ensure a smooth user experience.
    - **Solution:** Implemented real-time data synchronization using WebSocket or similar technologies to keep booking information consistent across all users. Added server-side validation to ensure data integrity.
3. **User Experience:**
    - **Challenge:** Creating a seamless and intuitive user interface required careful design and testing to ensure that users could easily navigate the application and complete their bookings.
    - **Solution:** Conducted user testing sessions and gathered feedback to refine the UI/UX design. Implemented responsive design techniques to ensure that the application was accessible on various devices.
4. **Cross-Browser Compatibility:**
    - **Challenge:** Ensuring that the application worked correctly across different web browsers and devices was essential for providing a consistent user experience.
    - **Solution:** Used cross-browser testing tools (e.g., BrowserStack) to test the application on multiple browsers and devices. Implemented polyfills and transpilers (e.g., Babel) to ensure compatibility with older browsers.
5. **Performance Optimization:**

    - **Challenge:** The application faced performance issues, such as slow load times and unresponsive interfaces, which impacted user satisfaction.
    - **Solution:** Implemented performance optimization techniques, including code minification, image compression, and asynchronous data loading. Used performance monitoring tools (e.g., Google Lighthouse) to identify and address bottlenecks.

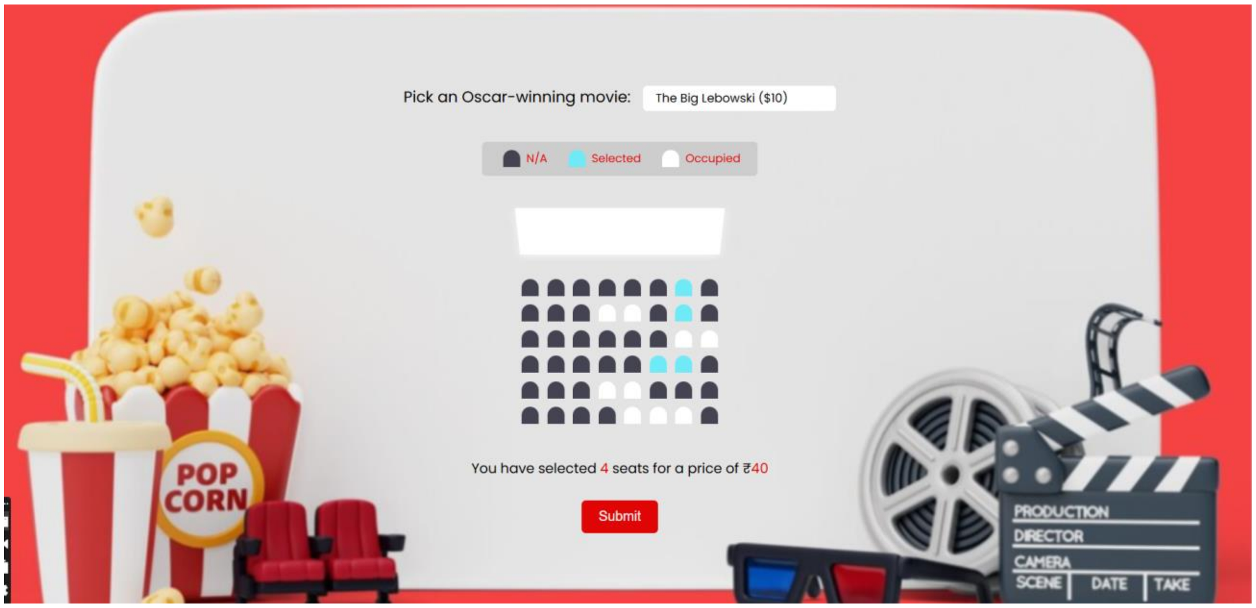
6. **Security:**
    - **Challenge:** Securing user data and preventing unauthorized access were critical for protecting sensitive information and maintaining user trust.
    - **Solution:** Implemented secure authentication methods (e.g., JWT) and encryption techniques to protect user data. Conducted regular security audits and applied best practices to mitigate common vulnerabilities.

Learning Outcomes:

1. **Proficiency in Web Development:**
    - Gained hands-on experience in creating dynamic web applications using HTML, CSS, and JavaScript. Developed skills in integrating frontend and backend components to build a complete ticket booking system.
2. **Database Management:**
    - Acquired knowledge in designing and managing databases to handle movie listings, showtimes, and bookings efficiently. Developed skills in performing CRUD operations and ensuring data integrity.
3. **UI/UX Design:**
    - Improved skills in designing user interfaces that are both visually appealing and functional. Learned to create responsive designs that enhance the user experience.
4. **Testing and Debugging:**
    - Developed strong debugging skills and an understanding of best practices for testing web applications. Gained experience in identifying and resolving issues to ensure application reliability.
5. **Deployment:**
    - Gained experience in deploying web applications to live environments and managing them post-deployment. Learned to handle server configurations and address any issues that arise.
6. **Problem-Solving:**

- Enhanced problem-solving skills by addressing challenges related to performance, data accuracy, and user experience. Developed solutions to improve the application's functionality and reliability.

Project interface:

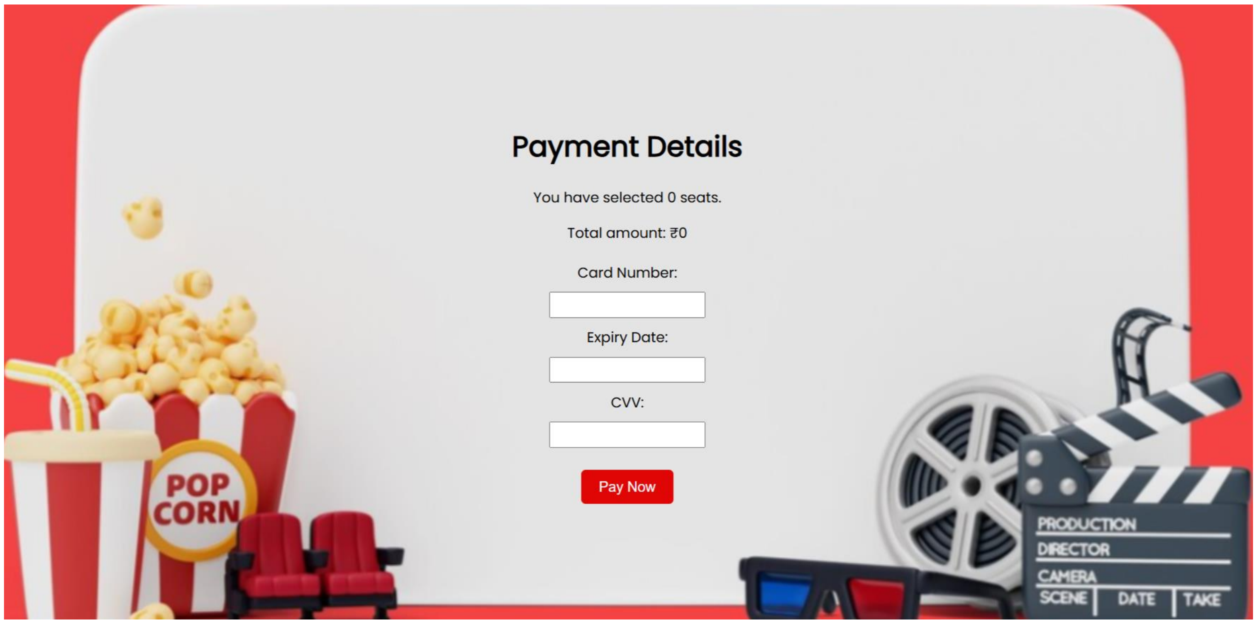


Pick an Oscar-winning movie: The Big Lebowski ($10)

N/A Selected Occupied

You have selected 4 seats for a price of ₹40

Payment Page:



Payment Details

You have selected 0 seats.

Total amount: ₹0

Card Number:

Expiry Date:

CVV:

Source code:

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <link rel="stylesheet" href="style.css" />
  <title>Movie Seat Booking</title>
</head>
<body>
  <div class="movie-container">
    <label for="movie">Pick an Oscar-winning movie:</label>
    <select name="movie" id="movie">
      <option value="10">The Big Lebowski ($10)</option>
      <option value="12">Fargo ($12)</option>
      <option value="8">O Brother ($8)</option>
      <option value="9">No Country for Old Men ($9)</option>
    </select>
  </div>
  <ul class="showcase">
    <li>
      <div class="seat"></div>
      <small>N/A</small>
    </li>
    <li>
      <div class="seat selected"></div>
      <small>Selected</small>
  </li>
  <li>
    <div class="seat occupied"></div>
    <small>Occupied</small>
  </li>
</ul>
<div class="container">
  <div class="screen"></div>
  <div class="row">
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
  </div>
  <div class="row">
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat occupied"></div>
    <div class="seat occupied"></div>
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
  </div>
  <div class="row">
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat occupied"></div>
    <div class="seat occupied"></div>
  </div>
  <div class="row">
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
  </div>
  <div class="row">
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat occupied"></div>
    <div class="seat occupied"></div>
    <div class="seat"></div>
    <div class="seat"></div>
    <div class="seat"></div>
  </div>
```

```
        <div class="row">
          <div class="seat"></div>
          <div class="seat"></div>
          <div class="seat"></div>
          <div class="seat"></div>
          <div class="seat occupied"></div>
          <div class="seat occupied"></div>
          <div class="seat occupied"></div>
          <div class="seat"></div>
        </div>
      </div>
      <p class="text">
        You have selected <span id="count">0</span> seats for a price of ₹<span id="total">0</span>
      </p>
      <button id="submit">Submit</button>
      <script src="script.js"></script>
    </body>
    </html>
```

<u>Style.css</u>

```
@import url("https://fonts.googleapis.com/css?family=Poppins&display=swap");

* {
  box-sizing: border-box;
}

body {
  background-color: #b1b0c2;
  background-image: url("Movie.png");
  background-size: cover;
  background-position: center;
  background-repeat: no-repeat;
  color: #fff;
  font-family: "Poppins", sans-serif;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  height: 100vh;
  margin: 0;
}

.movie-container {
  margin: 20px 0;
}

.movie-container label {
  color: #000; /* Black text for label */
  font-size: 18px;
}

.movie-container select {
  background-color: #fff;
  border: 0;
  border-radius: 5px;
  font-size: 14px;
  font-family: inherit;
  margin-left: 10px;
  padding: 5px 15px;

  -moz-appearance: none;

  -webkit-appearance: none;

  appearance: none;

}


.container {

  perspective: 1000px;

  margin-bottom: 30px;

}


.seat {

  background-color: #444451;

  height: 20px; /* Adjusted for better visibility */

  width: 20px; /* Adjusted for better visibility */

  margin: 5px; /* Increased margin */

  border-top-left-radius: 10px;

  border-top-right-radius: 10px;

  display: inline-block;

}


.seat.selected {

  background-color: #6feaf6;

}


.seat.occupied {

  background-color: #fff;

}
```

```
.seat:not(.occupied):hover {
  cursor: pointer;
  transform: scale(1.2);
}

.showcase {
  background: rgba(0, 0, 0, 0.1);
  padding: 5px 10px;
  border-radius: 5px;
  color: #df0606;
  list-style-type: none;
  display: flex;
  justify-content: space-between;
}

.showcase li {
  display: flex;
  align-items: center;
  justify-content: center;
  margin: 0 10px;
}

.showcase li small {
  margin-left: 2px;
}

.row {
  display: flex;
}
```

```
.screen {
  background-color: #fff;
  height: 70px;
  width: 100%;
  margin: 15px 0;
  transform: rotateX(-45deg);
  box-shadow: 0 3px 10px rgba(255, 255, 255, 0.7);
}

p.text {
  margin: 5px 0;
  color: #000; /* Black text color for the text paragraph */
}

p.text span {
  color: #df0606; /* Red color for the highlighted total amount */
}

#submit {
  background-color: #df0606; /* Red background for submit button */
  border: none;
  color: #fff; /* White text for submit button */
  padding: 10px 20px;
  font-size: 16px;
  border-radius: 5px;
  cursor: pointer;
  margin-top: 20px;
}
```

```
#submit:hover {
  background-color: #c10505; /* Darker red for hover effect */
}

.payment-container {
  text-align: center;
  padding: 20px;
}

.payment-container h1 {
  color: #000; /* Black color for the header */
}

.payment-container p {
  color: #000; /* Black color for paragraph text */
}

#payment-form {
  display: flex;
  flex-direction: column;
  align-items: center;
}

#payment-form label {
  margin: 5px 0;
  color: #000; /* Black color for form labels */
}
```

```css
#payment-form input {
   margin: 5px 0;
   padding: 5px;
}

#pay-button {
   background-color: #df0606; /* Red background for pay button */
   border: none;
   color: #fff; /* White text for pay button */
   padding: 10px 20px;
   font-size: 16px;
   border-radius: 5px;
   cursor: pointer;
   margin-top: 20px;
}

#pay-button:hover {
   background-color: #c10505; /* Darker red for hover effect */
}
```

Script.js

```javascript
const container = document.querySelector(".container");
const seats = document.querySelectorAll(".row .seat:not(.occupied)");
const count = document.getElementById("count");
const total = document.getElementById("total");
const movieSelect = document.getElementById("movie");
const submitButton = document.getElementById("submit");
let ticketPrice = +movieSelect.value;
```

```
function populateUI() {
  const selectedSeats = JSON.parse(localStorage.getItem("selectedSeats"));
  if (selectedSeats !== null && selectedSeats.length > 0) {
    seats.forEach((seat, index) => {
      if (selectedSeats.indexOf(index) > -1) seat.classList.add("selected");
    });
  }
  const selectedMovieIndex = localStorage.getItem("selectedMovieIndex");
  if (selectedMovieIndex !== null)
    movieSelect.selectedIndex = selectedMovieIndex;
}

function setMovieData(movieIndex, moviePrice) {
  localStorage.setItem("selectedMovieIndex", movieIndex);
  localStorage.setItem("selectedMoviePrice", moviePrice);
}

function updateSelectedCount() {
  const selectedSeats = document.querySelectorAll(".row .seat.selected");
  const seatsIndex = [...selectedSeats].map((seat) => [...seats].indexOf(seat));
  localStorage.setItem("selectedSeats", JSON.stringify(seatsIndex));
  const selectedSeatsCount = selectedSeats.length;
  count.innerText = selectedSeatsCount;
  total.innerText = selectedSeatsCount * ticketPrice;
}

movieSelect.addEventListener("change", (e) => {
  ticketPrice = +e.target.value;
  setMovieData(e.target.selectedIndex, e.target.value);
  updateSelectedCount();
});

container.addEventListener("click", (e) => {
  if (
    e.target.classList.contains("seat") &&
    !e.target.classList.contains("occupied")
  ) {
    e.target.classList.toggle("selected");
    updateSelectedCount();
  }
});

submitButton.addEventListener("click", (e) => {
  e.preventDefault(); // Prevent default form submission
  // Store the total amount and selected count in localStorage
  localStorage.setItem("selectedSeatsCount", count.innerText);
  localStorage.setItem("totalAmount", total.innerText);
  // Redirect to the payment page
  window.location.href = 'payment.html';
});

// Init
populateUI();
updateSelectedCount();
```

Payment.html

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

```
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <link rel="stylesheet" href="style.css" />
  <title>Payment Page</title>
</head>
<body>
  <div class="payment-container">
    <h1>Payment Details</h1>
    <p>You have selected <span id="selected-count">0</span> seats.</p>
    <p>Total amount: ₹<span id="total-amount">0</span></p>
    <form id="payment-form">
      <label for="card-number">Card Number:</label>
      <input type="text" id="card-number" name="card-number" required />
      <label for="expiry-date">Expiry Date:</label>
      <input type="text" id="expiry-date" name="expiry-date" required />
      <label for="cvv">CVV:</label>
      <input type="text" id="cvv" name="cvv" required />
      <button type="submit" id="pay-button">Pay Now</button>
    </form>
  </div>
  <script src="payment.js"></script>
</body>
</html>
```

Payment.js

```
document.getElementById("payment-form").addEventListener("submit", function (e) {
  e.preventDefault(); // Prevent default form submission
```

```
    // Simulate payment processing (you can add actual payment logic here)


    // Redirect to the booking portal after payment
    window.location.href = 'index.html';
});
```