**MERAFILM.COM**

PROJECT ON

**MERAFILM.COM**

Developed By

Name: Mohan Dass S Reg.No:S171163800343

**MERAFILM.COM**

Batch Code :S160923

Start Date : 26-12-2016 End Date: 06-04-2017

Name of the Coordinator: Maheswari N

Name of the Developer : Mohan Dass S

Date of Submission:27-02-2017

**CERTIFICATE**

This is to certified that report titled, MERAFILM.COM, embodies the original work done by in fulfillment of his course requirement at NIIT.

Coordinator: Maheswari N

**CONTENTS**

**CHAPTER NO TITLE PAGE.NO**

1. **INTRODUCTION 6**
   1. Company Profile 6
   2. Objective 7
2. **SYSTEM SPECIFICATION 5**
   1. Hardware Requirements 8
   2. Software Requirements 8
3. **SOFTWARE DESCRIPTION 9**
   1. Front End
   2. Middleware 13
   3. Back End 15
4. **PROJECT DESCRIPTION 24**
   1. Modules 24
   2. Modules Description 25
5. **SYSTEM TESTING 26**

5.1 Karma testing 26

1. **CONCLUSION 28**
   1. Conclusion 28
2. **APPENDIX 29**
   1. Source Code 29
   2. Screen Shots 74
3. **REFERENCES 81**

**CHAPTER 1**

**INTRODUCTION**

**1.1 COMPANY PROFILE**

NIIT, today a leader in global talent management, began as the brain child of two college students who had the vision of bringing people and computers together. In the early 1980’s NIIT established its first training academy to address the lack of trained professionals in India. From there, the school grew rapidly, and the organization built new tools to be better enable its campus, added development and consulting to its services to employ its new graduates, and new systems and service offerings.

Over the last three decades NIIT has development the skill set and expertise to be a trusted partner for companies and educational institutions around the world who want to make the most of their educational programs and systems. Research-based innovation remains a key driver for the company and has enabled it to develop programs and curricula that use cutting-edge instructional design methodologies and training delivery.

Today, NIIT offers training solutions in IT, Business Process Outsourcing, Banking, Finance and Insurance, Executive Management Education, and communication and Professional Life Skills, reaching more than five million learners every year, It operates more than 450 schools worldwide, offers turkey IT integration programs for schools and has provided computer-based learning to nearly 7.8 million students in more than 15,000 government and private schools. NIIT’s corporate Learning Solutions offers integrated learning solutions (including Managed Training Services, advisory services, learning content, development, technology, assessment, and the learning management systems) to Fortune 500 companies, Association Universities, Technology companies, Training corporations and publishing houses.

**1.2 OBJECTIVES**

An online movie booking which will manage end to end display the details of Movies and Theatres for MeraFilm.com. Customers can search for currently running movies, read reviews, watch trailers and also book the seats as per their choice. The admin should manage movies, show timings and reservations.

**Functional Requirements**

1. Provide role based access to system.
2. Application should be accessible over the Internet.
3. Customer to perform the following activities:
   1. Browse the movies based on show timings/ location
   2. Browse movies based on the theatre
   3. Read the reviews and give their rating and comments
   4. Book Tickets as per their choice.
4. Provide an authentication mechanism to verify the credentials of an employee assigned with the administrator role.
5. Provide an authorization mechanism to restrict access to administrative functions based on the fact whether the current user is an administrator or not.
6. An administrator should be able to perform the following activities:
   1. Add movie details for booking
   2. Delete movies from front page
   3. Manage seat availability from Back End. Random seat booking should be permitted.
   4. Explore theaters with selected movies.
   5. Add Show timings
   6. Store all details collected during reserving tickets in Database
   7. Appropriate reports with various filters to show details

**CHAPTER 2**

**SYSTEM SPECIFICATION**

**2.1 HARDWARE REQUIREMENTS**

Processor : Intel Core i3

RAM : 4 GB

Monitor : LCD 14.1’’

Keyboard : Compaq of 104 Keys

Mouse : Compaq

**2.2 SOFTWARE REQUIREMENTS**

Operating System : Windows 8.1

Front – End : Angular js.

Middle – End : Express js.

Back – End : Node js,Mongo.

**CHAPTER 3**

**SOFTWARE DESCRIPTION**

**3.1 FRONT END**

**Bootstrap:**

* Bootstrap is a free front-end framework for faster and easier web development
* Bootstrap includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels and many other, as well as optional JavaScript plugins
* Bootstrap also gives you the ability to easily create responsive designs

**CSS:**

* **CSS** stands for **C**ascading **S**tyle **S**heets
* CSS describes how HTML elements are to be displayed on screen, paper, or in other media
* CSS **saves a lot of work**. It can control the layout of multiple web pages all at once
* External style sheets are stored in **CSS files**

**JavaScript:**

* JavaScript is the programming language of HTML and the Web.
* **JavaScript** to program the behavior of web pages
* JavaScript is easy to learn.

**JQuery:**

* JQuery is a JavaScript Library.
* JQuery greatly simplifies JavaScript programming.
* jQuery is easy to learn.

**Angular JS:**

* AngularJS extends HTML with new attributes.
* AngularJS is perfect for Single Page Applications (SPAs).
* AngularJS is easy to learn.

**ANGULAR JSFRAMEWORK**

**AngularJS** is a complete JavaScript-based open-source front-end web application framework mainly maintained by Google and by a community of individuals and corporations to address many of the challenges encountered in developing single-page applications.

The JavaScript components complement Apache Cordova, the framework used for developing cross-platform mobile apps. It aims to simplify both the development and the testing of such applications by providing a framework for client-side model–view–controller (MVC) and model–view–viewmodel (MVVM) architectures, along with components commonly used in rich Internet applications.

**AngularJS's design goals include:**

* To decouple [DOM](https://en.wikipedia.org/wiki/Document_Object_Model) manipulation from application logic. The difficulty of this is dramatically affected by the way the code is structured.
* To decouple the client side of an application from the server side. This allows development work to progress in parallel, and allows for reuse of both sides.
* To provide structure for the journey of building an application from designing the UI, through writing the business logic, to testing.

**AngularJS** directives allow the developer to specify custom and reusable HTML-like elements and attributes that define data bindings and the behavior of presentation components.

Some of the most commonly used directives are:

**ng-app:**

Declares the root element of an AngularJS application, under which directives can be used to declare bindings and define behavior.

**ng-bind:**

Sets the text of a DOM element to the value of an expression. For example, <span ng-bind="name"></span> displays the value of ‘name’ inside the span element. Any change to the variable ‘name’ in the application's scope reflect instantly in the DOM.

**ng-model:**

Similar to ng-bind, but establishes a two-way data binding between the view and the scope.

**ng-model-options:**

Provides tuning for how model updates are done.

**ng-controller:**

Specifies a JavaScript controller class that evaluates HTML expressions.

**ng-repeat:**

Instantiate an element once per item from a collection.

**ng-show&ng-hide:**

Conditionally show or hide an element, depending on the value of a boolean expression. Show and hide is achieved by setting the CSS display style.

**ng-switch:**

Conditionally instantiate one template from a set of choices, depending on the value of a selection expression.

**ng-view:**

The base directive responsible for handling routes[[12]](https://en.wikipedia.org/wiki/AngularJS#cite_note-12) that resolve JSON before rendering templates driven by specified controllers.

**ng-if:**

Basic if statement directive that instantiates the following element if the conditions are true. When the condition is false, the element is removed from the DOM. When true, a clone of the compiled element is re-inserted.

**TWO WAY DATA BINDING:**

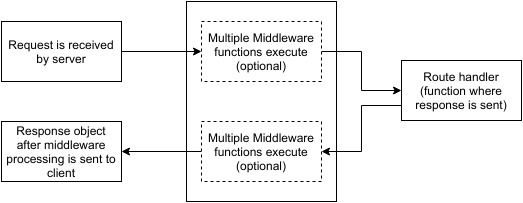
AngularJS two-way data binding is its most notable feature, largely relieving the server backend of templating responsibilities. Instead, templates are rendered in plain HTML according to data contained in a scope defined in the model. The $scope service in Angular detects changes to the model section and modifies HTML expressions in the view via a controller.

**MIDDLEWARE**

Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications. It is an open source framework developed and maintained by the Node.js foundation.

Middleware functions are functions that have access to the request object (req), the response object (res), and the next middleware function in the application’s request-response cycle. These functions are used to modify req and res objects for tasks like parsing request bodies, adding response headers, etc.

The following diagram summarizes what we have learnt about middleware:



* **BODY-PARSER**

This is used to parse the body of requests which have payloads attached to them. To mount body parser, we need to install it using npm install --save body-parser and to mount it, include the following lines in index.js:

varbodyParser=require('body-parser');

//To parse URL encoded data

app.use(bodyParser.urlencoded({ extended:false}))

//To parse json data

app.use(bodyParser.json())

* **COOKIE-PARSER**

It parses Cookie header and populate req.cookies with an object keyed by cookie names. To mount cookie parser, we need to install it using npm install --save cookie-parser and to mount it, include the following lines in index.js:

varbodyParser=require('body-parser');

app.use(cookieParser())

* **EXPRESS-SESSION**

It creates a session middleware with the given options. We will discuss its usage in the Sessions section.

**BACK END**

**NODE JS:**

Node.js is a cross-platform environment and library for running JavaScript applications which is used to create networking and server-side applications.

# Features of Node.js

* **Extremely fast:**

Node.js is built on Google Chrome's V8 JavaScript Engine, so its library is very fast in code execution.

* **I/O is Asynchronous and Event Driven:**

All APIs of Node.js library are asynchronous i.e. non-blocking. So a Node.js based server never waits for an API to return data. The server moves to the next API after calling it and a notification mechanism of Events of Node.js helps the server to get a response from the previous API call. It is also a reason that it is very fast.

* **Single threaded:**

Node.js follows a single threaded model with event looping.

* **Highly Scalable:**

Node.js is highly scalable because event mechanism helps the server to respond in a non-blocking way.

* **No buffering:**

Node.js cuts down the overall processing time while uploading audio and video files. Node.js applications never buffer any data. These applications simply output the data in chunks.

* **Open source:**

Node.js has an open source community which has produced many excellent modules to add additional capabilities to Node.js applications.

* **License:**

Node.js is released under the MIT license.

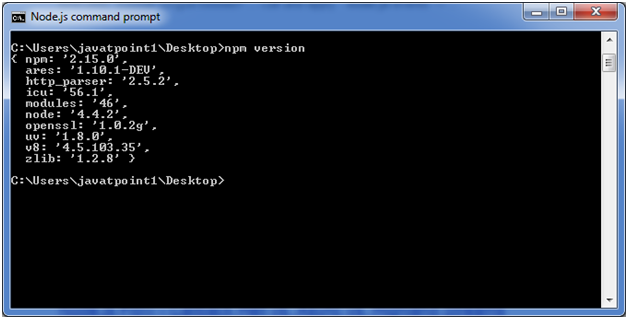
# Node.js Installation

Node Package Manager provides two main functionalities:

* It provides online repositories for node.js packages/modules which are searchable on search.nodejs.org
* It also provides command line utility to install Node.js packages, do version management and dependency management of Node.js packages.

The npm comes bundled with Node.js installables in versions after that v0.6.3. we can check the version by opening Node.js command prompt and typing the following command:

1. npm  version



## Installing Modules using npm

Following is the syntax to install any Node.js module:

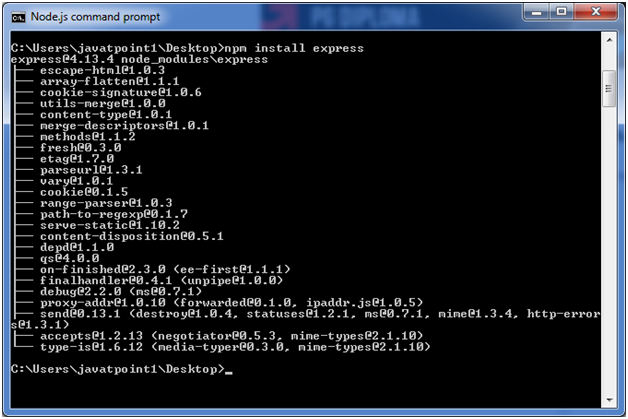
* npm install **<MODULE** Name**>**

Let's install a famous Node.js web framework called express:

Open the Node.js command prompt and execute the following command:

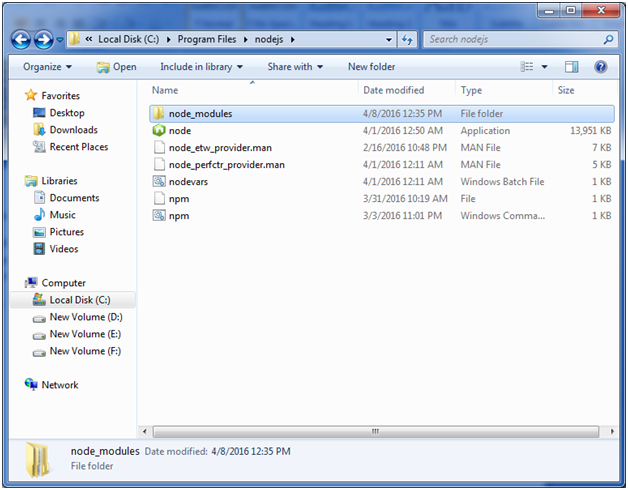
* npm install express

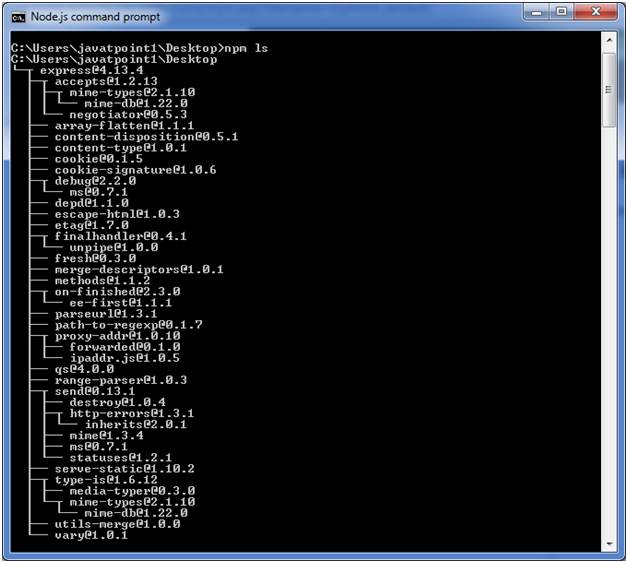
we can see the result after installing the "express" framework.



## Global vs Local Installation:

By default, npm installs dependency in local mode. Here local mode specifies the folder where Node application is present. For example if you installed express module, it created node\_modules directory in the current directory where it installed express module.

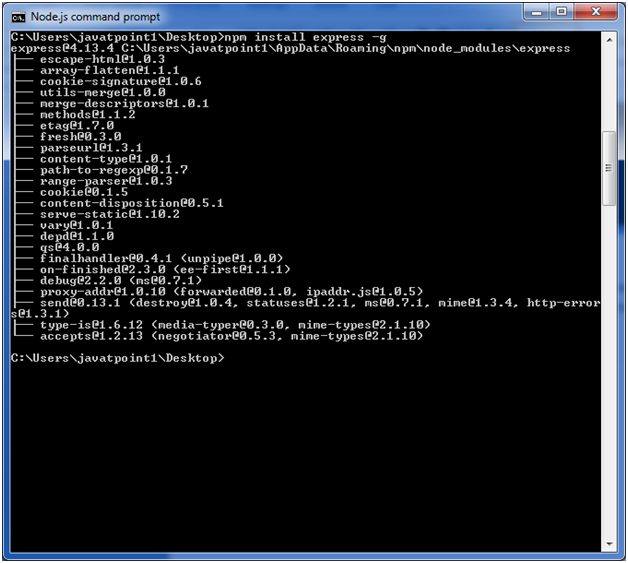




Globally installed packages/dependencies are stored in system directory. Let's install express module using global installation. Although it will also produce the same result but modules will be installed globally.

Open Node.js command prompt and execute the following code:

* npm install express -g

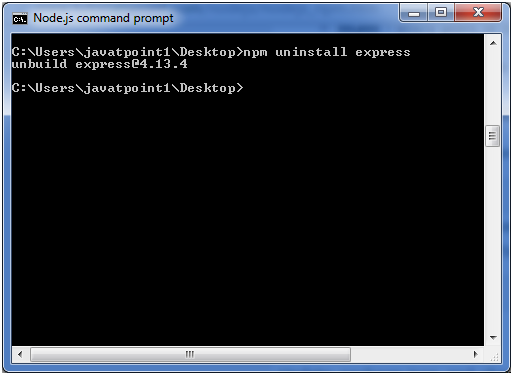
;

Here first line tells about the module version and its location where it is getting installed.

## Uninstalling a Module:

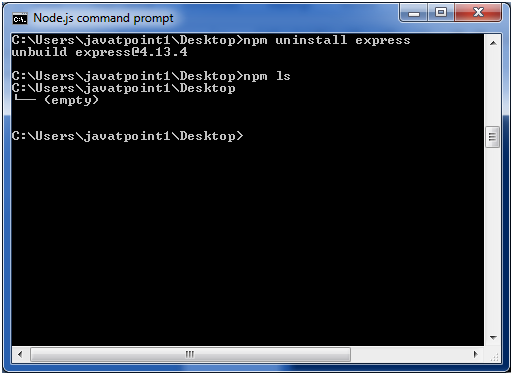
To uninstall a Node.js module, use the following command:

* npm uninstall express



The Node.js module is uninstalled. You can verify by using the following command:

* npm ls



we can see that the module is empty now.

## Searching a Module

"npm search express" command is used to search express or module.

* npm search express

**CHAPTER 4**

**PROJECT DESCRIPTION**

**4.1 MODULES**

1. Admin

1.1 Show Time Assigning

1.2 City Assigning

1.3 Theater Assigning

1.4 Movie Time Assigning

1.5 Mapping

2. User

2.1 Booking Page

2.2 Confirmation Page

2.3 Review page

**4.2 MODULE DESCRIPTION**

**4.2.1 Admin :**

1. **Assign Show Time**

In this module admin can add, edit and delete Show Time

1. **Assign City**

In this module, admin can add, edit and delete Supplier City details.

1. **Assign Theaters**

In this module, admin can add, edit and delete Theater information like Theater Name, Show Time, and Location

1. **Assign Movies**

In this module, admin can add, edit and delete Movie information. These information are getting from OMDB api.

1. **mapping**

In this module, admin can add, edit and delete mapping informations.

**4.2.2 User:**

1. **Booking Page**

This module can allow a user to book movies by clicking the

particular movie.

1. **Confirmation Page**

In this module, user can view the Confirmation details of the booked Movie

1. **Rating Page**

In this module, User can rate the Movie by clicking the particular movie.

**CHAPTER 5**

**SYSTEM TESTING**

Testing is a process of checking whether the developed system is working according to the original objectives and requirements. Testing is a set of activities that can be planned in advance and conducted systematically. Testing is vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct, the global will be successfully achieved. In adequate testing if not testing leads to error that may not appear even many months. This creates two problems,

* The time lag between the cause and the appearance of the problem.
* The effect of the system errors on the files and records within the system.

A Small system error can collectively explode into much larger problem. Effective testing early in the purpose translates directly into long term cost savings from a reduced number of errors. Another reason for system testing is its utility, as a user-oriented vehicle before implementation. The best programs are worthless if it produces the correct outputs.

**5.1 Karma Testing**

Karma is essentially a tool which spawns a web server that executes source code against test code for each of the browsers connected. The results of each test against each browser are examined and displayed via the command line to the developer such that they can see which browsers and tests passed or failed.

A browser can be captured either

manually, by visiting the URL where the Karma server is listening (typically http://localhost:9876/),

or automatically by letting Karma know which browsers to start when Karma is run .

Karma also watches all the files, specified within the configuration file, and whenever any file changes, it triggers the test run by sending a signal to the testing server to inform all of the captured browsers to run the test code again. Each browser then loads the source files inside an IFrame, executes the tests and reports the results back to the server.

The server collects the results from all of the captured browsers and presents them to the developer.

**CHAPTER 6**

**6.1 Conclusion**

In general, today’s businesses must always strive to create the next best thing that consumers will want because consumers continue to desire their services. To continuously be better and faster.  In this world of new technology, businesses need to accommodate to the new types of consumer needs and trends because it will prove to be vital to their business’ success and survival.

From the inception of the Internet and BookingWebsite, the possibilities have become endless for has its both businesses and consumers.  However, just like anything else, MovieBookingWebsite disadvantages including consumer uncertainties, but nothing that cannot be resolved or avoided by good decision-making and business practices.

**CHAPTER 7**

**APPENDIX**

* 1. **SOURCE CODE**

**Front End:**

**MovieBook.html:**

|  |
| --- |
| <div class="container"> |
|  |  |
|  |  |
|  |  |
|  | <img src="{{movieinfo.moviPoster}}" style="width:300px;height:400px;float: left;"> |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | <div class="table-responsive"> |
|  | <table class="table"> |
|  | <tr> |
|  | <td><b>Movie title</b></td> |
|  | <td> |
|  | <p>{{movieinfo.moviTitle}}</p> |
|  | </td> |
|  | <!-- <td>ng-model="book.FilmName"{{movieinfo.moviTitle}} </td> --> |
|  | </tr> |
|  |  |
|  | <tr> |
|  | <td><b>Select City</b></td> |
|  | <td><select id="select" ng-model="book.CityName" ng-change="sel()"> |
|  | <option ng-repeat="l in locMovie" value="{{l}}" >{{l}}</option> |
|  | </select></td> |
|  | <!-- <td><select ng-blur="e()" ng-model="book.CityName"> |
|  | <option id="cityNames"></option> |
|  | </select></td> --> |
|  | </tr> |
|  | <tr> |
|  | <td><b>Select Theater</b></td> |
|  | <td><select ng-model="book.HallName"> |
|  | <option ng-repeat="t in loctheat" value="{{t}}">{{t}}</option> |
|  | </select></td> |
|  | </tr> |
|  | <tr> |
|  | <td><b>Select Date</td> |
|  | <td> <input ng-blur="d()" type="date" id="datebook"/></td> |
|  | </tr> |
|  | <tr> |
|  | <td><b>Select Show Time</b></td> |
|  | <td> |
|  | <select ng-model="book.ShowTime" ng-change="showseat()"> |
|  | <option ng-repeat="map in maplist|filter:movieinfo.moviTitle |filter:book.HallName">{{map.ShowTime}}</option> |
|  | </select></td> |
|  | </tr> |
|  | <tr> |
|  | <td><b>Amount</b></td> |
|  | <td> |
|  | <select ng-model="book.Amount"> |
|  | <option>50</option> |
|  | <option>100</option> |
|  | <option>150</option> |
|  | </select></td> |
|  |  |
|  | </tr> |
|  |  |
|  | <tr> |
|  | <td><b>Select Number Of Tickets</b></td> |
|  | <td> |
|  | <select ng-model="book.NoTickets"> |
|  | <option>1</option> |
|  | <option>2</option> |
|  | <option>3</option> |
|  | <option>4</option> |
|  | <option>5</option> |
|  |  |
|  | </select></td> |
|  |  |
|  | </tr> |
|  |  |
|  |  |
|  |  |
|  | <tr> |
|  | <td> |
|  | <!-- <p><button id="generate">Generate</button></p> --> |
|  | <!-- <p><code id="output"></code></p> --> |
|  | </td> |
|  | <td> |
|  | <button ng-click="add()" class="btn btn-primary">Done</button> |
|  | </td> |
|  | </tr> |
|  | <!-- |
|  | <tr>Book |
|  | <td><b>total Amount</b></td> |
|  | <td><p>{{Totalamount}}</p></td> |
|  | </tr> --> |
|  |  |
|  | </table> |
|  | </div> |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | <div ng-show="seat"> |
|  | <table class="seatLayout"style="margin-top:1%;"> |
|  |  |
|  | <div style="margin-left:5%;margin-top:7%;"> |
|  | <img src="img/seat2.gif" style="width:30px;height:30px;">Available |
|  | <img src="img/seat1.gif" style="width:30px;height:30px;">Reserved |
|  | </div> |
|  | <tbody> |
|  | <tr> |
|  | <td></td> |
|  | <td ng-repeat="col in cols track by $index" ng-bind="col"></td> |
|  | </tr> |
|  | <tr class="tablestyle" ng-repeat="row in rows"> |
|  | <td ng-bind="row"></td> |
|  | <td ng-repeat="col in cols" ng-click="seatClicked(row + col)" ng-switch on="getStatus(row + col)" ng-disabled="disableSeat" style="padding: 5px 5px 5px 5px;"> |
|  | <img ng-switch-when="selected" src="img/seat1.gif" style="width:25px;height:25px;"> |
|  | <img ng-switch-when="reserved" src="img/seat1.gif" style="width:25px;height:25px;"> |
|  | <img ng-switch-default src="img/seat2.gif" style="width:25px;height:25px;"> |
|  | <!-- <img id="myImage" ng-switch-default src="img/seat2.gif" style="width:25px;height:25px;"> --> |
|  |  |
|  | </td> |
|  | </tr> |
|  | </tbody> |
|  | </table> |
|  |  |
|  | <div style="background-color:black;width:31%;"> |
|  | <h4 style="color:white;text-align:center">SCREEN</h4> </div> |
|  | <div class="table-responsive"> |
|  | <table class="table"> |
|  | <tr> |
|  | <td><b>Seating Arrangement</b></td> |
|  | <td><p id="seatno"></p></td> |
|  | </tr> |
|  |  |
|  | <tr> |
|  | <td><b>No Of seats</b></td> |
|  | <td><p>{{NumberOfSeats}}</p></td> |
|  | </tr><p><code id="output"></code></p> |
|  | <!-- <tr> |
|  | <td><b>booking id</b></td> |
|  | <td><p><code id="output"></code></p></td> |
|  | </tr> --> |
|  | <tr> |
|  | <td> <button ng-click="bk(Booklist)" class="btn btn-primary">Confirm</button></td> |
|  | </tr> |
|  | </table> |
|  | </div> |
|  | </div> |
|  | </div> |

**Confirm.html:**

|  |
| --- |
| <div class="container"> |
|  | <div class="table-responsive"> |
|  | <table class="table"> |
|  | <tr> |
|  | <td><b>BOOKING ID</b></td> |
|  | <td> |
|  | <p>{{confirmInfo.bookingid}}</p> |
|  | </td> |
|  | </tr> |
|  |  |
|  | <tr> |
|  | <td><b>Movie title</b></td> |
|  | <td> |
|  | <p>{{confirmInfo.Title}}</p> |
|  | </td> |
|  | </tr> |
|  | <tr> |
|  | <td><b>City</b></td> |
|  | <td> |
|  | <p>{{confirmInfo.CityName}}</p> |
|  | </td> |
|  | </tr> |
|  |  |
|  | <tr> |
|  | <td><b>Theater</b></td> |
|  | <td> |
|  | <p>{{confirmInfo.HallName}}</p> |
|  | </td> |
|  | </tr> |
|  |  |
|  | <tr> |
|  | <td><b>Date</b></td> |
|  | <td> |
|  | <p>{{confirmInfo.Day}}</p> |
|  | </td> |
|  | </tr> |
|  |  |
|  | <tr> |
|  | <td><b>Show Time</b></td> |
|  | <td> |
|  | <p>{{confirmInfo.ShowTime}}</p> |
|  | </td> |
|  | </tr> |
|  |  |
|  | <tr> |
|  | <td><b>no of seats</b></td> |
|  | <td> |
|  | <p>{{confirmInfo.NoTickets}}</p> |
|  | </td> |
|  |  |
|  | <tr> |
|  | <td><b>Seat Number</b></td> |
|  | <td> |
|  | <p>{{confirmInfo.seatnumbers}}</p> |
|  | </td> |
|  | </tr> |
|  | <tr> |
|  | <td><b>User Name</b></td> |
|  | <td> |
|  | <input type="text" ng-model="confirm.cnUser"> |
|  | </td> |
|  | </tr> |
|  | <tr> |
|  | <td><b>Mail ID</b></td> |
|  | <td> |
|  | <input type="email" ng-model="confirm.cnMail"> |
|  | </td> |
|  | </tr> |
|  | <tr> |
|  | <td><b>ACNT.NO</b></td> |
|  | <td> |
|  | <input type="number"> |
|  | </td> |
|  | </tr> |
|  | <tr> |
|  | <td><b>CDN</b></td> |
|  | <td> |
|  | <input type="string"> |
|  | </td> |
|  | </tr> |
|  | <tr> |
|  | <td><b>Total Amount</b></td> |
|  | <td> |
|  | <p>{{confirmInfo.totalAmnt}}</p> |
|  | </td> |
|  | </tr> |
|  |  |
|  |  |
|  |  |
|  |  |
|  | <tr> |
|  |  |
|  | <td> |
|  | <td><button ng-click="ConfirmBook(confirmlist)" class="btn btn-success">CONFIRM</button></td> |
|  | </td> |
|  | </tr> |
|  |  |
|  |  |
|  | </table> |
|  | </div> |

**Admin.html:**

|  |
| --- |
| <style> |
|  | .nav-side-menu { |
|  | font-size: 12px; |
|  | overflow: auto; |
|  | font-weight: 200; |
|  | /\*background-color: #000;\*/ |
|  | top: 0px; |
|  | width: 100%; |
|  | height: 100%; |
|  | color: #e1ffff; |
|  | margin-top:15px; |
|  | overflow: hidden; |
|  | } |
|  | .nav-side-menu .brand { |
|  | background-color: #23282e; |
|  | line-height: 38px; |
|  | display: none; |
|  | text-align: center; |
|  | font-size: 14px; |
|  | } |
|  | .nav-side-menu .toggle-btn { |
|  | display: none; |
|  |  |
|  | } |
|  | .nav-side-menu ul, |
|  | .nav-side-menu li { |
|  | list-style: none; |
|  | padding: 0px; |
|  | margin: 0px; |
|  | line-height: 35px; |
|  | cursor: pointer; |
|  | font-size: 13px; |
|  | } |
|  | .nav-side-menu ul a > strong, |
|  | .nav-side-menu li > a > strong{ |
|  | font-size: 1.0em; |
|  | font-weight: 400; |
|  | vertical-align: middle; |
|  | } |
|  | .nav-side-menu ul :not(collapsed) .arrow:before, |
|  | .nav-side-menu li :not(collapsed) .arrow:before { |
|  | font-family: FontAwesome; |
|  | /\*content: "\f078";\*/ |
|  | content:"\f107"; |
|  | display: inline-block; |
|  | padding-left: 10px; |
|  | padding-right: 10px; |
|  | vertical-align: middle; |
|  | float: right; |
|  | font-size:14px; |
|  | } |
|  | .nav-side-menu ul .active, |
|  | .nav-side-menu li .active { |
|  | /\*border-left: 3px solid #fe0266; |
|  | background-color: #4f5b69;\*/ |
|  | } |
|  | .nav-side-menu ul .sub-menu li.active, |
|  | .nav-side-menu li .sub-menu li.active { |
|  | color: black; |
|  | } |
|  | .nav-side-menu ul .sub-menu li.active a, |
|  | .nav-side-menu li .sub-menu li.active a { |
|  | color:#1182c5; |
|  | } |
|  | .nav-side-menu ul .sub-menu li, |
|  | .nav-side-menu li .sub-menu li { |
|  | /\*background-color: #181c20;\*/ |
|  | border: none; |
|  | line-height: 28px; |
|  | /\*border-bottom: 1px solid #23282e;\*/ |
|  | margin-left: 0px; |
|  | } |
|  | .nav-side-menu ul .sub-menu li:hover, |
|  | .nav-side-menu li .sub-menu li:hover { |
|  | /\*background-color: #020203;\*/ |
|  | } |
|  | .nav-side-menu ul .sub-menu li:before, |
|  | .nav-side-menu li .sub-menu li:before { |
|  | font-family: FontAwesome; |
|  | content: "\f105"; |
|  | /\*content:"\f107";\*/ |
|  | display: inline-block; |
|  | padding-left: 10px; |
|  | padding-right: 10px; |
|  | vertical-align: middle; |
|  | /\*color:#333;\*/ |
|  | font-weight:normal; |
|  | } |
|  | .nav-side-menu li { |
|  | padding-left: 0px; |
|  | /\*border-left: 3px solid #fe0266; |
|  | border-bottom: 1px solid #23282e;\*/ |
|  | } |
|  | .nav-side-menu li a { |
|  | text-decoration: none; |
|  | color: #333; |
|  | } |
|  | .nav-side-menu li a:hover { |
|  | text-decoration: none; |
|  | color: #1182c5; |
|  | } |
|  | .nav-side-menu li a i { |
|  | display: inline-block; |
|  | margin-right: 5px; |
|  | vertical-align: middle; |
|  | width: 32px; |
|  | } |
|  | .nav-side-menu li:hover { |
|  | /\*border-left: 3px solid #1182c5; |
|  | background-color: #4f5b69;\*/ |
|  | -webkit-transition: all 1s ease; |
|  | -moz-transition: all 1s ease; |
|  | -o-transition: all 1s ease; |
|  | -ms-transition: all 1s ease; |
|  | transition: all 1s ease; |
|  | } |
|  | @media (max-width: 767px) { |
|  | .nav-side-menu { |
|  | position: relative; |
|  | width: 100%; |
|  | margin-bottom: 10px; |
|  | } |
|  | .nav-side-menu .toggle-btn { |
|  | display: block; |
|  | cursor: pointer; |
|  | position: absolute; |
|  | right: 10px; |
|  | top: 10px; |
|  | z-index: 10 !important; |
|  | padding: 3px; |
|  | background-color: #ffffff; |
|  | color: #000; |
|  | width: 40px; |
|  | text-align: center; |
|  | } |
|  | .nav-side-menu .brand { |
|  | text-align: left !important; |
|  | padding-left: 20px; |
|  | line-height: 50px !important; |
|  | display:block; |
|  | } |
|  | } |
|  | @media (min-width: 767px) { |
|  | .nav-side-menu .menu-list .menu-content { |
|  | display: block; |
|  | } |
|  | } |
|  |  |
|  |  |
|  | /\*--------------------profile-------------------\*/ |
|  | .fb-profile{ |
|  | position:relative; |
|  | } |
|  | .fb-profile img.fb-image-lg{ |
|  | z-index: 0; |
|  | width: 100%; |
|  | margin-bottom: 10px; |
|  | } |
|  |  |
|  | .fb-image-profile |
|  | { |
|  | margin:-50px 10px 0 50px; |
|  | z-index: 9; |
|  | width: 10%; |
|  | } |
|  | .fb-profile-text{ |
|  | position:relative; |
|  | margin-bottom: 50px; |
|  | z-index: 1; |
|  | } |
|  | .fb-profile-text h4{ |
|  | margin-bottom:4px; |
|  | display:inline-block; |
|  | vertical-align:middle; |
|  | } |
|  | .fb-profile-text p{ |
|  | margin-bottom:4px; |
|  | display:inline-block; |
|  | vertical-align:middle; |
|  | } |
|  | .thumbnail{ |
|  | border-radius: 15px; |
|  | } |
|  | .profile-imgbox{ |
|  | bottom:0px; |
|  | left: auto; |
|  | position: absolute; |
|  | right: 0; |
|  | text-align: center; |
|  | z-index: 2; |
|  | } |
|  | .profile-imgbox ul{ |
|  | list-style:none; |
|  | margin:0; |
|  | padding:0; |
|  | } |
|  | .profile-imgbox ul li{ |
|  | list-style:none; |
|  | margin:0; |
|  | padding:0; |
|  | display:inline-block; |
|  | text-align:center; |
|  | vertical-align:bottom; |
|  | } |
|  | .profile-imgbox ul li a{ |
|  | display:block; |
|  | padding:0 18px; |
|  | text-align:center; |
|  | text-decoration:none; |
|  |  |
|  | } |
|  | .profile-imgbox ul li a span{ |
|  | background:#0088CB; |
|  | border: 2px solid #ffffff; |
|  | border-radius: 4px; |
|  | display: block; |
|  | margin: 0 auto; |
|  | padding: 10px; |
|  | text-align: center; |
|  | width: 100px; |
|  | } |
|  | .profile-imgbox ul li a span i { |
|  | display:inline-block; |
|  | color:#FFF; |
|  | vertical-align: middle; |
|  |  |
|  | } |
|  | .profile-imgbox ul li a b{ |
|  | display:block; |
|  | font-weight:normal; |
|  | height:48px; |
|  | } |
|  | .profile-imgbox p{ |
|  | text-align:center; |
|  | display:block; |
|  | font-size:18px; |
|  | margin-bottom:10px; |
|  | background:rgba(0,0,0,0.44); |
|  | display:inline-block; |
|  | padding:4px 24px; |
|  | border-radius:4px; |
|  | color:#ffffff; |
|  | } |
|  | .project-detail{ |
|  | } |
|  | .project-detail h4{ |
|  | color: #202020; |
|  | font-size: 24px; |
|  | font-weight: 700; |
|  | margin: 0; |
|  | text-transform: uppercase; |
|  | } |
|  | .project-detail b{ |
|  | background: #202020 ; |
|  | display: inline-block; |
|  | text-align: center; |
|  | height: 1px; |
|  | width: 8%; |
|  | } |
|  | .project-detailBox{ |
|  | margin-bottom:18px; |
|  | margin-top:18px; |
|  | } |
|  |  |
|  | </style> |
|  |  |
|  | <div class="container"> |
|  | <div class="row"> |
|  |  |
|  | <div class="col-sm-12 col-md-12"> |
|  |  |
|  | <div class="nav-side-menu"> |
|  | <div class="brand">Filter</div> |
|  | <button aria-expanded="true" data-target="#menu-content" data-toggle="collapse" class="fa fa-bars fa-2x toggle-btn"></button> |
|  |  |
|  | <div class="menu-list"> |
|  |  |
|  | <ul class="menu-content collapse in" id="menu-content"> |
|  |  |
|  |  |
|  | <li class="collapsed active" data-target="#showdetail" data-toggle="collapse"> |
|  | <span class="glyphicon glyphicon-time" style="color:blue"></span> <strong style="color:black">ASSIGN SHOW TIMES</strong> |
|  | </li> |
|  |  |
|  |  |
|  |  |
|  | <ul id="showdetail" class="sub-menu collapse"> |
|  | <li class="active"> <table class="table"> |
|  | <thead> |
|  | <tr> |
|  |  |
|  | <th>Show time</th> |
|  |  |
|  |  |
|  | </tr> |
|  | </thead> |
|  | <tbody> |
|  | <tr> |
|  |  |
|  |  |
|  | <td><input ng-model="tim.showTime"/></td> |
|  |  |
|  | <td><button ng-click="addShow()" class="btn btn-primary">Add Time</button></td> |
|  | <td><button ng-click="updateShow()" class="btn btn-primary">Update</button></td> |
|  |  |
|  | </tr> |
|  | <tr ng-repeat="tim in timlist"> |
|  |  |
|  | <td>{{tim.showTime}}</td> |
|  |  |
|  | <td><button ng-click="removeShow(tim)" class="btn btn-danger">Remove</button></td> |
|  | <td><button ng-click="editShow(tim)" class="btn btn-primary">Edit</button></td> |
|  | </tr> |
|  | </tbody> |
|  | </table></li> |
|  | </ul> |
|  |  |
|  |  |
|  | <li class="collapsed" data-target="#citydetail" data-toggle="collapse"> |
|  | <!-- <a href="#"> --> |
|  | <span class="glyphicon glyphicon-map-marker" style="color:red"></span> <strong style="color:black">ASSIGN THEATERS </strong></a> |
|  | </li> |
|  |  |
|  | <ul id="citydetail" class="sub-menu collapse"> |
|  | <li class="active"> <table class="table"> |
|  | <thead> |
|  | <tr> |
|  |  |
|  | <th>City Name</th> |
|  |  |
|  |  |
|  | </tr> |
|  | </thead> |
|  | <tbody> |
|  | <tr> |
|  |  |
|  |  |
|  | <td><input ng-model="loc.cityName"/></td> |
|  |  |
|  | <td><button ng-click="addCity()" class="btn btn-primary">Add City</button></td> |
|  | <td><button ng-click="updateCity()" class="btn btn-primary">Update</button></td> |
|  |  |
|  | </tr> |
|  | <tr ng-repeat="loc in loclist"> |
|  |  |
|  | <td>{{loc.cityName}}</td> |
|  |  |
|  | <td><button ng-click="removeCity(loc)" class="btn btn-danger">Remove</button></td> |
|  | <td><button ng-click="editCity(loc)" class="btn btn-primary">Edit</button></td> |
|  | </tr> |
|  | </tbody> |
|  | </table></li> |
|  | </ul> |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | <li class="collapsed" data-target="#theaterdetail" data-toggle="collapse"> |
|  | <!-- <a href="#"> --> |
|  | <span class="glyphicon glyphicon-equalizer" style="color:#73C422"></span> <strong style="color:black">ASSIGN THEATERS</strong></a> |
|  | </li> |
|  | <ul id="theaterdetail" class="sub-menu collapse"> |
|  | <li class="active"> <table class="table"> |
|  | <thead> |
|  | <tr> |
|  |  |
|  | <th>Theater Name</th> |
|  | <th>City</th> |
|  | <th>Show time</th> |
|  |  |
|  | </tr> |
|  | </thead> |
|  | <tbody> |
|  | <tr> |
|  |  |
|  |  |
|  | <td><input ng-model="thtre.TheaterName"/></td> |
|  | <td> |
|  | <div class="form-group"> |
|  |  |
|  | <select class="form-control" ng-model="thtre.location"> |
|  | <option ng-repeat="loc in loclist">{{loc.cityName}}</option> |
|  |  |
|  | </select></div></td> |
|  |  |
|  | <td> |
|  | <div> |
|  | <select id="selecttime" class="form-control" > |
|  | <option ng-repeat="tim in timlist">{{tim.showTime}}</option> |
|  |  |
|  | </select> |
|  | </div> |
|  | </td> |
|  | <td> |
|  | <div> |
|  |  |
|  | <td><button id="add" class="btn btn-primary">Add Time</button></td> |
|  | </div> |
|  | <div> |
|  | <td><select id="time" ng-model="thtre.showTime" multiple ></select></td> |
|  |  |
|  | </div></td> |
|  |  |
|  |  |
|  |  |
|  | <td><button ng-click="addTheater()" class="btn btn-primary">Add Theater</button></td> |
|  | <td><button ng-click="updateTheater()" class="btn btn-primary">Update</button></td> |
|  |  |
|  | </tr> |
|  | <tr ng-repeat="thtre in thtrelist"> |
|  |  |
|  | <td>{{thtre.TheaterName}}</td> |
|  | <td>{{thtre.location}}</td> |
|  | <td>{{thtre.showTime}}</td> |
|  |  |
|  | <td><button ng-click="removeTheater(thtre)" class="btn btn-danger">Remove</button></td> |
|  | <td><button ng-click="editTheater(thtre)" class="btn btn-primary">Edit</button></td> |
|  | </tr> |
|  | </tbody> |
|  | </table></li></ul> |
|  |  |
|  |  |
|  |  |
|  | <li class="collapsed" data-target="#moviedetail" data-toggle="collapse"> |
|  | <!-- <a href="#"> --> |
|  | <span class="glyphicon glyphicon-facetime-video" style="color:#0A378B"></span> <strong style="color:black">ASSIGN MOVIES</strong> </a> |
|  | </li> |
|  | <ul id="moviedetail" class="sub-menu collapse fade"> |
|  | <li class="active"> |
|  | <form class="container" class="table"> |
|  | <table> |
|  | <tbody> |
|  | <tr> |
|  | <td>Movie Name<input ng-model="movi.moviTitle" placeholder="Title"/></td> |
|  | <td><button ng-click="addMovie(movi)" class="btn btn-primary">Add </button></td> |
|  |  |
|  | <td><button ng-click="updateMovie()" class="btn btn-primary">Update</button></td></tr> |
|  |  |
|  | <!-- </tbody> --> |
|  |  |
|  | <tr ng-repeat="m in moviList"> |
|  | <td>{{m.moviTitle}}</td> |
|  | <td>{{m.moviYear}}</td> |
|  | <td>{{m.moviLanguage}}</td> |
|  | <!-- <td>{{m.moviPoster}}</td> --> |
|  | <td><img src="{{m.moviPoster}}" style="width:100px;height:100px;"></td> |
|  | <!-- <td>{{movi.moviePlot}}</td> |
|  | <td>{{movi.moviePoster}}</td> --> |
|  | <td><button ng-click="removeMovie(m)" class="btn btn-danger">Remove</button></td> |
|  | <td><button ng-click="editMovie(m)" class="btn btn-primary">Edit</button></td> |
|  | </tr> |
|  | </tbody> |
|  | </table> |
|  |  |
|  |  |
|  |  |
|  |  |
|  | </form> |
|  | </li> |
|  |  |
|  | <!-- <li><a href="#">Carpenter</a></li> --> |
|  | </ul> |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | <li class="collapsed" data-target="#mapping" data-toggle="collapse"> |
|  | <!-- <a href="#"> --> |
|  | <span class="glyphicon glyphicon-blackboard" style="color:#C926A2"></span> <strong style="color:black">MAPPING</strong> </a> |
|  | </li> |
|  | <ul id="mapping" class="sub-menu collapse"> |
|  | <li class="active"> <table class="table"> |
|  | <thead> |
|  | <tr> |
|  | <th>Movie</th> |
|  | <th>City</th> |
|  | <th>Theater</th> |
|  | <th>Show time</th> |
|  |  |
|  | <th>From</th> |
|  | <th>To</th> |
|  |  |
|  | </tr> |
|  | </thead> |
|  | <tbody> |
|  | <tr> |
|  |  |
|  | <td> |
|  | <div class="form-group"> |
|  |  |
|  | <select class="form-control" ng-model="map.Film"> |
|  | <option ng-repeat="m in moviList">{{m.moviTitle}}</option> |
|  |  |
|  | </select></div></td> |
|  |  |
|  | <td> |
|  | <div class="form-group"> |
|  |  |
|  | <select class="form-control" ng-model="map.City"> |
|  | <option ng-repeat="loc in loclist">{{loc.cityName}}</option> |
|  | </select></div></td> |
|  | <td> |
|  | <div class="form-group"> |
|  |  |
|  | <select id="thname" class="form-control" ng-model="map.Hall" > |
|  |  |
|  | <option ng-repeat="th in thtrelist |filter:map.City">{{th.TheaterName}}</option> |
|  | </select></div></td> |
|  | <select class="form-control" id="selecttime" ng-model="map.ShowTime"> |
|  | <optgroup label="Select" ng-repeat="t in thtrelist| filter:map.Hall"> |
|  | <option ng-repeat="t in t.showTime" value={{t}}>{{t}}</option> |
|  | </optgroup> |
|  | </select> |
|  |  |
|  |  |
|  | </td> |
|  | </td> --> |
|  |  |
|  | <td><input type="date" ng-model="map.From"></td> |
|  | <td><input type="date" ng-model="map.To"></td> |
|  |  |
|  |  |
|  |  |
|  |  |
|  | <td><button ng-click="addMap()" class="btn btn-primary">Add</button></td> |
|  | <td><button ng-click="updateMap()" class="btn btn-primary">Update</button></td> |
|  |  |
|  | </tr> |
|  | <tr ng-repeat="map in maplist"> |
|  | <td>{{map.Film}}</td> |
|  | <td>{{map.City}}</td> |
|  | <td>{{map.Hall}}</td> |
|  | <td>{{map.ShowTime}}</td> |
|  | <td>{{map.From}}</td> |
|  | <td>{{map.To}}</td> |
|  |  |
|  | <td><button ng-click="removeMap(map)" class="btn btn-danger">Remove</button></td> |
|  | <td><button ng-click="editMap(map)" class="btn btn-primary">Edit</button></td> |
|  | </tr> |
|  | </tbody> |
|  | </table></li></ul> |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | </div> |
|  | </div> |

**ticketBookingController.js:**

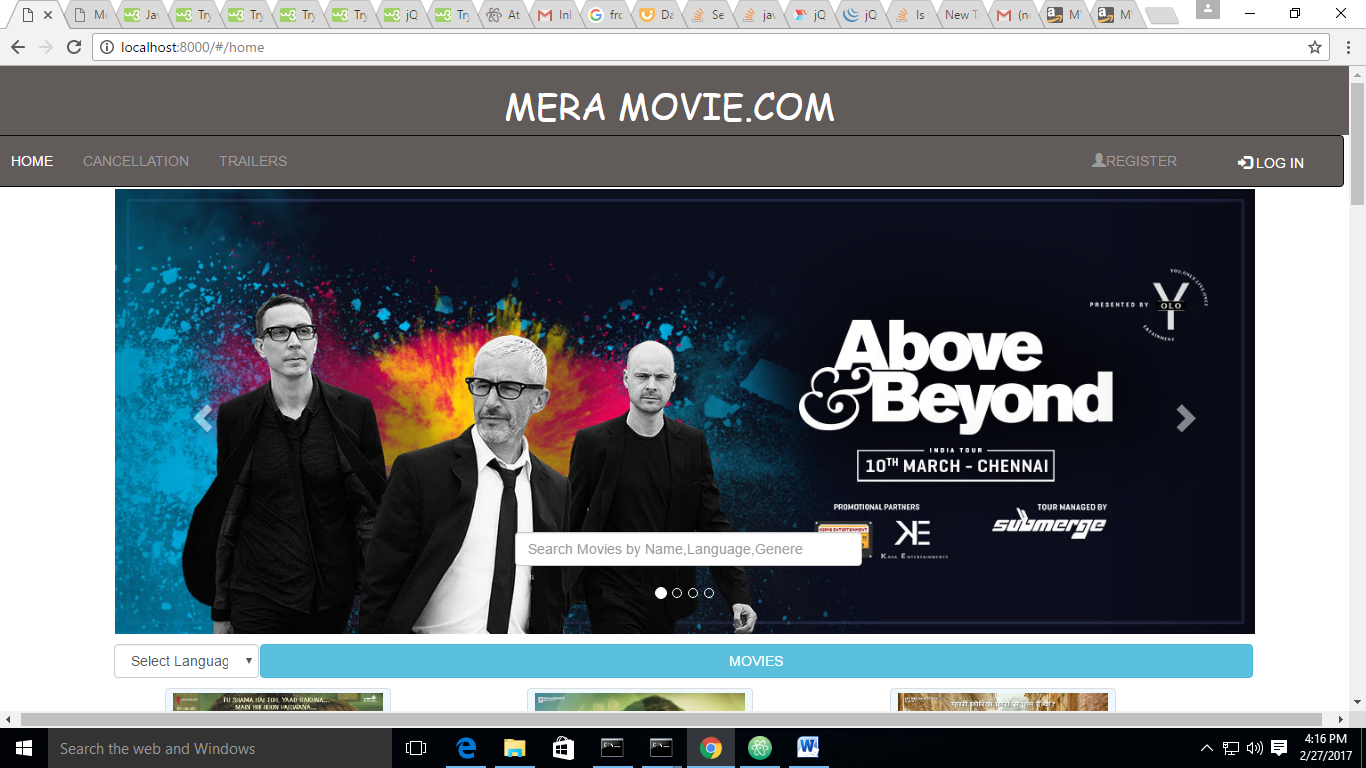
|  |
| --- |
| 'use strict'; |
|  |  |
|  | module.exports = function($scope, $http,$log, $rootScope,$location) { |
|  | $scope.movieinfo = $rootScope.bookedMovie; |
|  | var date; |
|  | var details=[]; |
|  | $scope.seat=false; var i; |
|  | $rootScope.seatArrange=[]; |
|  | console.log( $scope.movieinfo ); |
|  |  |
|  | // document.getElementById("datebook").value=date; |
|  |  |
|  |  |
|  | var refreshBookin = function () { |
|  | $http.get('/bok/bok').success(function (response) { |
|  | console.log('book READ IS SUCCESSFUL'); |
|  | $scope.Booklist = response; |
|  | $scope.book = ""; |
|  | }); |
|  | }; |
|  |  |
|  | refreshBookin(); |
|  |  |
|  |  |
|  |  |
|  |  |
|  | // |
|  | // $scope.rt1=function(){ |
|  | // var value=1; |
|  | // return value; |
|  | // } |
|  |  |
|  | var refreshConfirm = function () { |
|  |  |
|  | $http.get('/con/con').success(function (response) { |
|  | console.log('Confirm READ IS SUCCESSFUL'); |
|  | $scope.confirmlist = response; |
|  | $scope.confirm = ""; |
|  | }); |
|  | }; |
|  |  |
|  | refreshConfirm(); |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | // |
|  | var selected=[]; |
|  | var reserved=[]; |
|  |  |
|  | $scope.rows = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J']; |
|  | $scope.cols = [1, 2, 3, 4, 5, 6, 7, 8 ,9 ,10 ]; |
|  |  |
|  | $scope.getStatus = function(seatPos) { |
|  | if(reserved.indexOf(seatPos) > -1) { |
|  | return 'reserved'; |
|  | } else if(selected.indexOf(seatPos) > -1) { |
|  | return 'selected'; |
|  | } |
|  |  |
|  | } |
|  |  |
|  |  |
|  |  |
|  | $scope.seatClicked=function(seatPos){ |
|  | var index = selected.indexOf(seatPos); |
|  | if(index != -1) { |
|  | // seat already selected, remove |
|  | selected.splice(index, 1) |
|  | } else { |
|  | // new seat, push |
|  | selected.push(seatPos); |
|  | console.log(selected); |
|  | // $rootScope.seatArrange=selected; |
|  | // console.log($rootScope.seatArrange); |
|  |  |
|  |  |
|  | document.getElementById("seatno").innerHTML=selected; |
|  | // document.getElementById('myImage').src="img/seat1.gif"; |
|  | // selected=document.getElementById("seatno").innerHTML; |
|  | $scope.NumberOfSeats=selected.length; |
|  | // $scope.book.seatnumbers=selected; |
|  | $scope.book.totalAmnt=$scope.book.Amount\*$scope.NumberOfSeats; |
|  |  |
|  |  |
|  | } |
|  | }; |
|  |  |
|  | $scope.d=function(){ |
|  | date=document.getElementById("datebook").value; |
|  | // alert(date); |
|  | console.log(date); |
|  | } |
|  |  |
|  |  |
|  |  |
|  |  |
|  | $scope.add =function() |
|  | { |
|  | // $scope.seat = true; |
|  |  |
|  | $scope.book.Day=date; |
|  | $scope.book.Title=$scope.movieinfo.moviTitle; |
|  | console.log($scope.book.Title); |
|  | console.log($scope.book.CityName); |
|  | console.log($scope.book.HallName); |
|  | console.log($scope.book.Day); |
|  |  |
|  | console.log($scope.book.ShowTime); |
|  |  |
|  |  |
|  | // selectdate = document.getElementById("date").value; |
|  | // console.log(selectdate); |
|  | // console.log($scope.book.ShowTime); |
|  |  |
|  | // details=$scope.confirmlist; |
|  | try |
|  | { |
|  | for(i=0;i<=$scope.confirmlist.length;i++) |
|  | { |
|  | if($scope.confirmlist.length==0) |
|  | { |
|  | $scope.seat = true; |
|  | } |
|  | else{ |
|  | console.log($scope.confirmlist[i].Title); |
|  | console.log($scope.confirmlist[i].cnCityName); |
|  | console.log($scope.confirmlist[i].cnHallName); |
|  | console.log($scope.confirmlist[i].cnDay); |
|  | console.log($scope.confirmlist[i].cnShowTime); |
|  |  |
|  |  |
|  | // console.log($scope.confirmlist[i].Title); |
|  |  |
|  | if ($scope.confirmlist[i].Title==$scope.book.Title && $scope.confirmlist[i].cnCityName==$scope.book.CityName && $scope.confirmlist[i].cnHallName==$scope.book.HallName && $scope.confirmlist[i].cnDay==$scope.book.Day && $scope.confirmlist[i].cnShowTime==$scope.book.ShowTime) |
|  | { |
|  | $scope.seat = true; |
|  | console.log($scope.confirmlist[i].Title); |
|  | console.log($scope.confirmlist[i].cnCityName); |
|  | console.log($scope.confirmlist[i].cnHallName); |
|  | console.log($scope.confirmlist[i].cnDay); |
|  | console.log($scope.confirmlist[i].cnShowTime); |
|  |  |
|  | // reserved.push($scope.confirmlist[i].cnseatnumbers); |
|  | // selected.push($scope.confirmlist[i].cnseatnumbers); |
|  |  |
|  | reserved=$scope.confirmlist[i].cnseatnumbers; |
|  | console.log(reserved); |
|  |  |
|  | } |
|  | else |
|  | { |
|  | $scope.seat = true; |
|  | } |
|  | } |
|  | } |
|  |  |
|  | } |
|  | catch(e){} |
|  |  |
|  | }; |
|  |  |
|  |  |
|  | // |
|  | // $scope.touch=function(Booklist){ |
|  | // $scope.book.Title=$scope.movieinfo.moviTitle; |
|  | // $http.post('/bok/bok', $scope.book).success(function (response) { |
|  | // console.log(response); |
|  | // |
|  | // }); |
|  | // refreshBookin(); |
|  | // } |
|  |  |
|  |  |
|  |  |
|  | /////////////////////////////////////////////////// |
|  | $scope.bk = function (Booklist) { |
|  | var arr = []; |
|  | while(arr.length < 1){ |
|  | var randomnumber = Math.ceil(Math.random()\*100000) |
|  | if(arr.indexOf(randomnumber) > -1) continue; |
|  | arr[arr.length] = randomnumber; |
|  | } |
|  | $scope.book.bookingid= arr; |
|  | console.log($scope.book.bookingid); |
|  | $scope.book.Title=$scope.movieinfo.moviTitle; |
|  | $scope.book.seatnumbers=selected; |
|  | // $scope.book.seatnumber= |
|  | console.log($scope.book); |
|  | $http.post('/bok/bok', $scope.book).success(function (response) { |
|  | console.log(response); |
|  |  |
|  | }); |
|  | $rootScope.confirmPage=$scope.book; |
|  | $location.path('/Confirm'); |
|  | refreshBookin(); |
|  |  |
|  | }; |
|  |  |
|  |  |
|  |  |
|  | var refreshLocat = function () { |
|  | $http.get('/cty/cty').success(function (response) { |
|  | console.log('READ IS SUCCESSFUL'); |
|  | $scope.loclist = response; |
|  | $scope.loc = ""; |
|  |  |
|  |  |
|  |  |
|  | }); |
|  | }; |
|  |  |
|  | refreshLocat(); |
|  | var uniqueNames = []; |
|  | var uniqueObj = []; |
|  | var uniqueHall=[]; |
|  | var uniqueTheat=[]; |
|  |  |
|  |  |
|  | var refreshMape = function () { |
|  | $http.get('/map/map').success(function (response) { |
|  | console.log('READ IS SUCCESSFUL'); |
|  | $scope.maplist = response; |
|  | console.log($scope.maplist); |
|  | $scope.map = ""; |
|  |  |
|  |  |
|  | for(i = 0; i< $scope.maplist.length; i++){ |
|  | if($scope.maplist[i].Film==$scope.movieinfo.moviTitle){ |
|  | if(uniqueNames.indexOf($scope.maplist[i].City) === -1){ |
|  | uniqueObj.push($scope.maplist[i]); |
|  | uniqueNames.push($scope.maplist[i].City); |
|  |  |
|  | // if($scope.maplist[i].City==$scope.book.CityName){ |
|  | // if(uniqueNames.indexOf($scope.maplist[i].City) === -1){ |
|  | // uniqueObj.push($scope.maplist[i]); |
|  | // uniqueNames.push($scope.maplist[i].City); |
|  | // } |
|  | // }} |
|  | } |
|  | }} |
|  | console.log(uniqueNames); |
|  |  |
|  | console.log($scope.locMovie); |
|  | console.log($scope.book.CityName); |
|  |  |
|  |  |
|  |  |
|  | console.log(uniqueNames); |
|  | console.log(uniqueNames); |
|  |  |
|  | console.log($scope.locMovie); |
|  |  |
|  | }); |
|  |  |
|  | }; |
|  | $scope.locMovie=uniqueNames; |
|  | $scope.loctheat=uniqueHall; |
|  |  |
|  | refreshMape(); |
|  |  |
|  | $scope.sel=function(){ |
|  | var j; |
|  | for( j= 0; j< $scope.maplist.length; j++){ |
|  |  |
|  | if($scope.maplist[j].Film==$scope.movieinfo.moviTitle&&$scope.maplist[j].City==$scope.book.CityName){ |
|  | if(uniqueHall.indexOf($scope.maplist[j].Hall) === -1){ |
|  | uniqueTheat.push($scope.maplist[j]); |
|  | uniqueHall.push($scope.maplist[j].Hall); |
|  | console.log(uniqueHall); |
|  | }} |
|  | } |
|  | } |
|  | var refreshTheat = function () { |
|  | $http.get('/theater/theater').success(function (response) { |
|  | console.log('theater READ IS SUCCESSFUL'); |
|  | $scope.thtrelist = response; |
|  | $scope.thtre = ""; |
|  |  |
|  | }); |
|  | } |
|  |  |
|  | refreshTheat(); |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | }; |

**Booking-crud.js:**

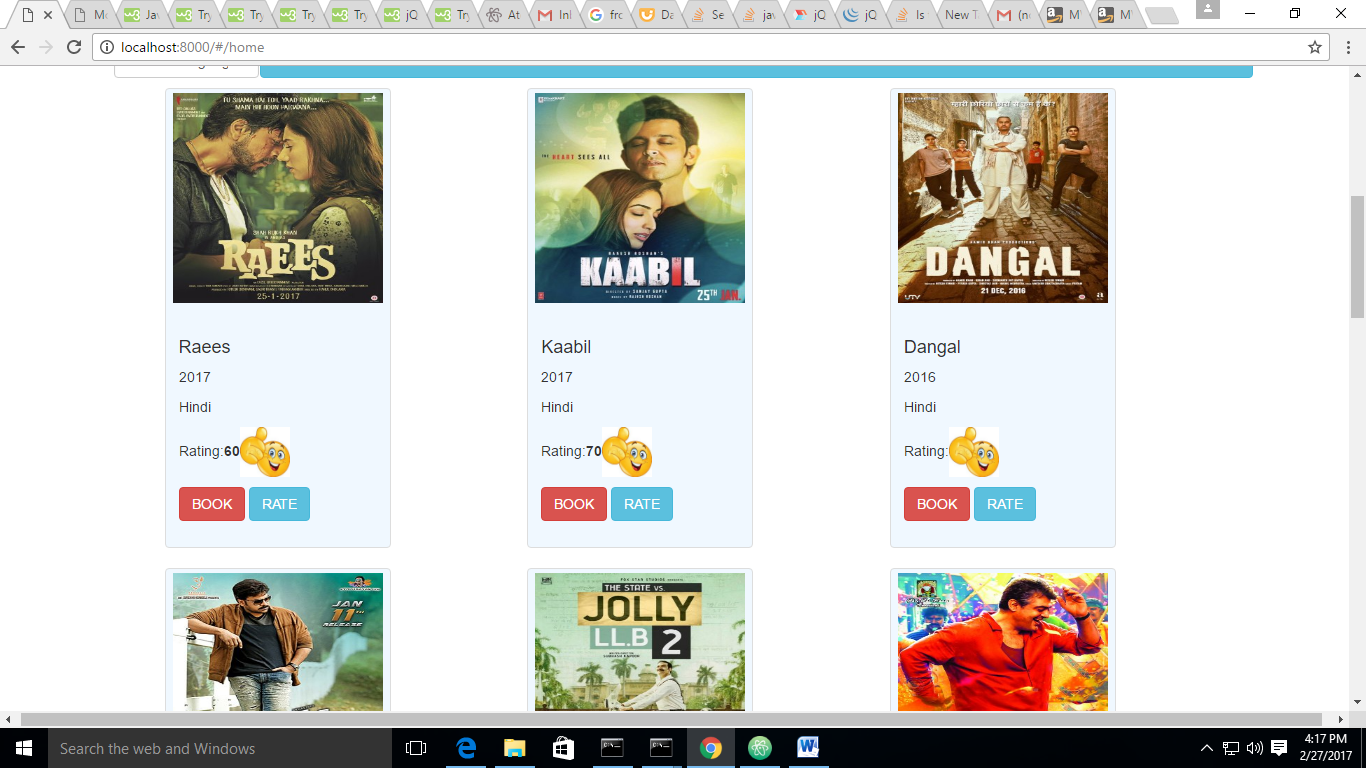
|  |
| --- |
| ar express = require('express'); |
|  | var router = express.Router(); |
|  | var bodyParser = require('body-parser'); |
|  |  |
|  |  |
|  | router.use(bodyParser.urlencoded({ extended: true })); |
|  |  |
|  |  |
|  | var mongoose = require('mongoose'); |
|  |  |
|  | var bookSchema = mongoose.Schema({ |
|  | bookingid:String, |
|  | Title:String, |
|  | CityName:String, |
|  | HallName:String, |
|  | Day:String, |
|  | ShowTime:String, |
|  | Amount:Number, |
|  | totalAmnt:String, |
|  | NoTickets:Number, |
|  | seatnumbers:String |
|  | }); |
|  | var Book = mongoose.model('Book',bookSchema, 'bookTable'); |
|  |  |
|  |  |
|  | router.get('/bok', function (req, res) { |
|  | console.log("REACHED GET FUNCTION ON Booking SERVER"); |
|  | Book.find({}, function (err, docs) { |
|  | res.json(docs); |
|  |  |
|  | }); |
|  | }); |
|  |  |
|  | router.get('/bok/:id', function (req, res) { |
|  | console.log("REACHED GET ID FUNCTION ON booking SERVER"); |
|  | Book.find({\_id: req.params.id}, function (err, docs) { |
|  | res.json(docs); |
|  |  |
|  | }); |
|  | }); |
|  |  |
|  | router.post('/bok', function(req, res){ |
|  | console.log(req.body); |
|  | var filmname = req.body.Title; |
|  | var cityname= req.body.CityName; |
|  | var theatname = req.body.HallName; |
|  | var dayname= req.body.Day; |
|  | var showname = req.body.ShowTime; |
|  | var moneyamnt = req.body.Amount; |
|  | var Quantity = req.body.NoTickets; |
|  | var seat = req.body.seatnumbers; |
|  |  |
|  | var book1 = new Book({ |
|  | Title:filmname, |
|  | CityName:cityname, |
|  | HallName:theatname, |
|  | Day:dayname, |
|  | ShowTime:showname, |
|  | Amount:moneyamnt, |
|  | NoTickets:Quantity, |
|  | seatnumbers:seat |
|  | }); |
|  |  |
|  | book1.save(function(err, docs){ |
|  | if ( err ) throw err; |
|  | console.log("booking Saved Successfully"); |
|  | res.json(docs); |
|  | }); |
|  |  |
|  | }) |
|  |  |
|  | router.delete('/bok/:id', function(req, res){ |
|  | console.log("REACHED Delete FUNCTION ON SERVER"); |
|  | Book.remove({\_id:req.params.id}, function(err, docs){ |
|  | res.json(docs); |
|  | }); |
|  | }) |
|  |  |
|  | router.put('/bok/:id', function(req, res){ |
|  | console.log("REACHED PUT"); |
|  | console.log(req.body); |
|  | Book.findOneAndUpdate({\_id:req.params.id}, req.body, function (err, data) { |
|  | res.json(data); |
|  | }); |
|  | }) |
|  |  |
|  | router.use(function(req, res, next) { |
|  | var err = new Error('Not Found'); |
|  | err.status = 404; |
|  | next(err); |
|  | }); |
|  |  |
|  | module.exports = router; |

* 1. **SCREEN SHOTS**

**Home Page**



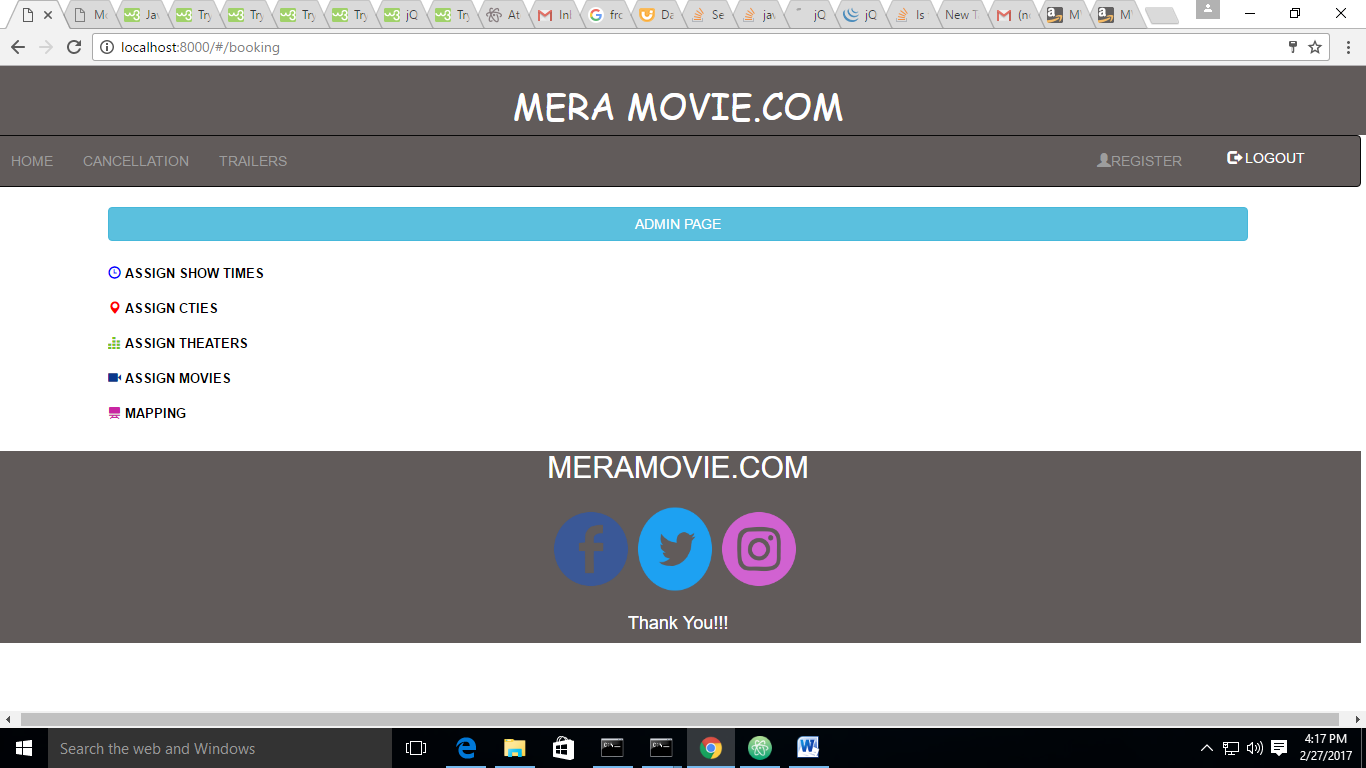
**Login Page**



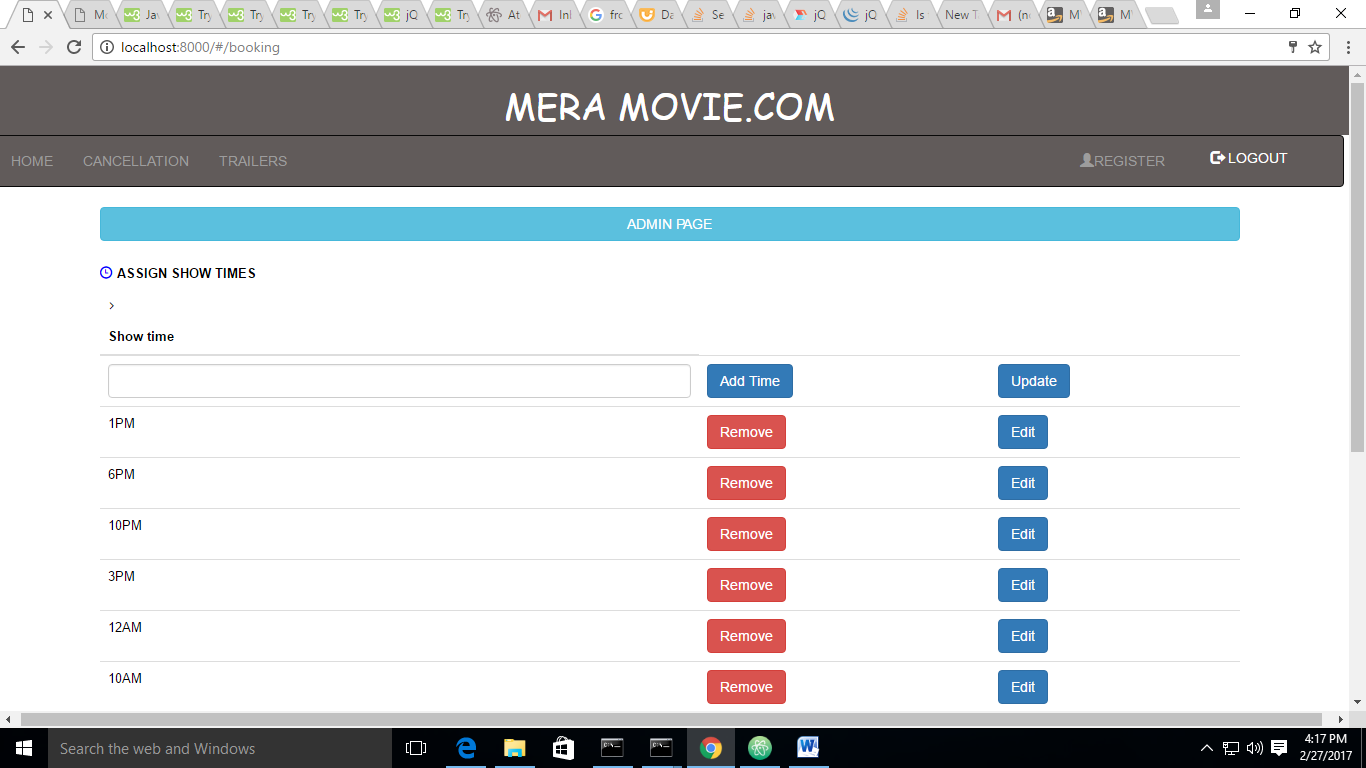
**Registration Page**



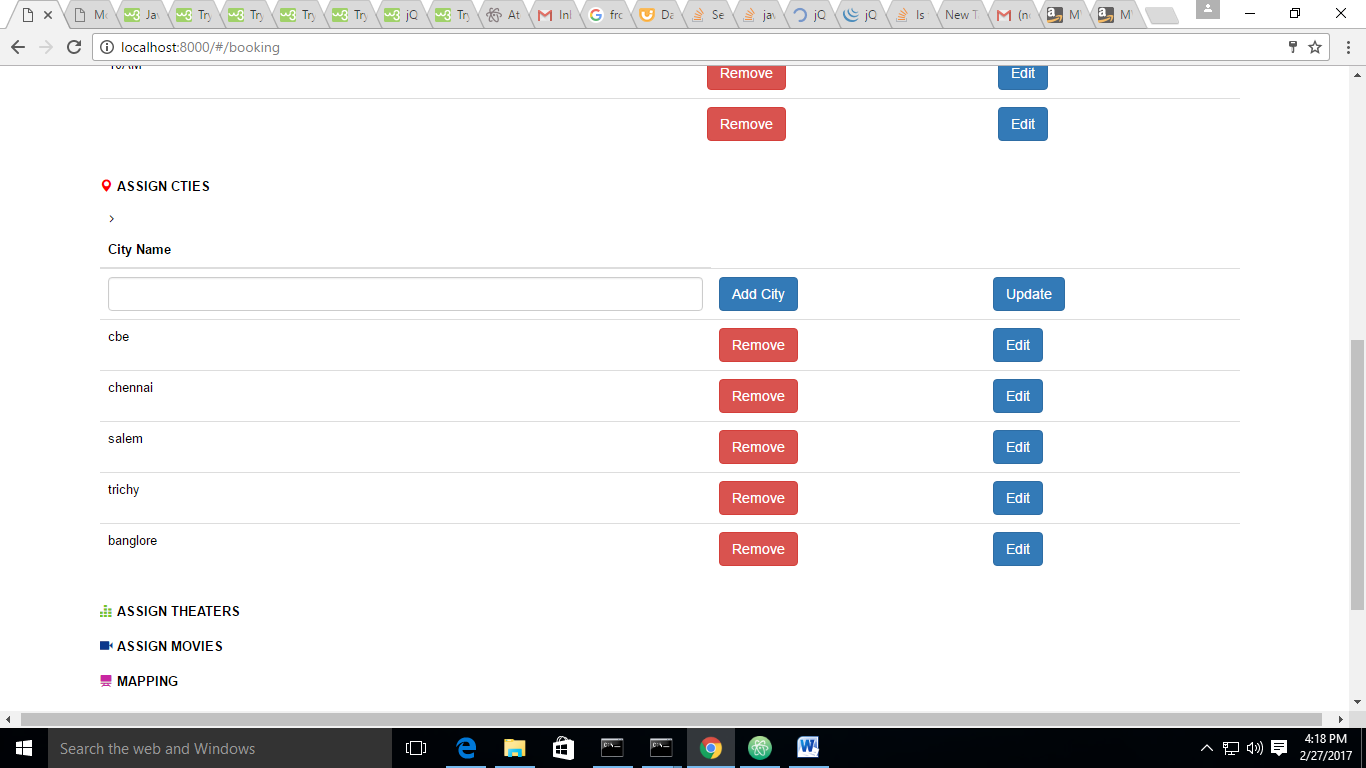
**Admin Page**



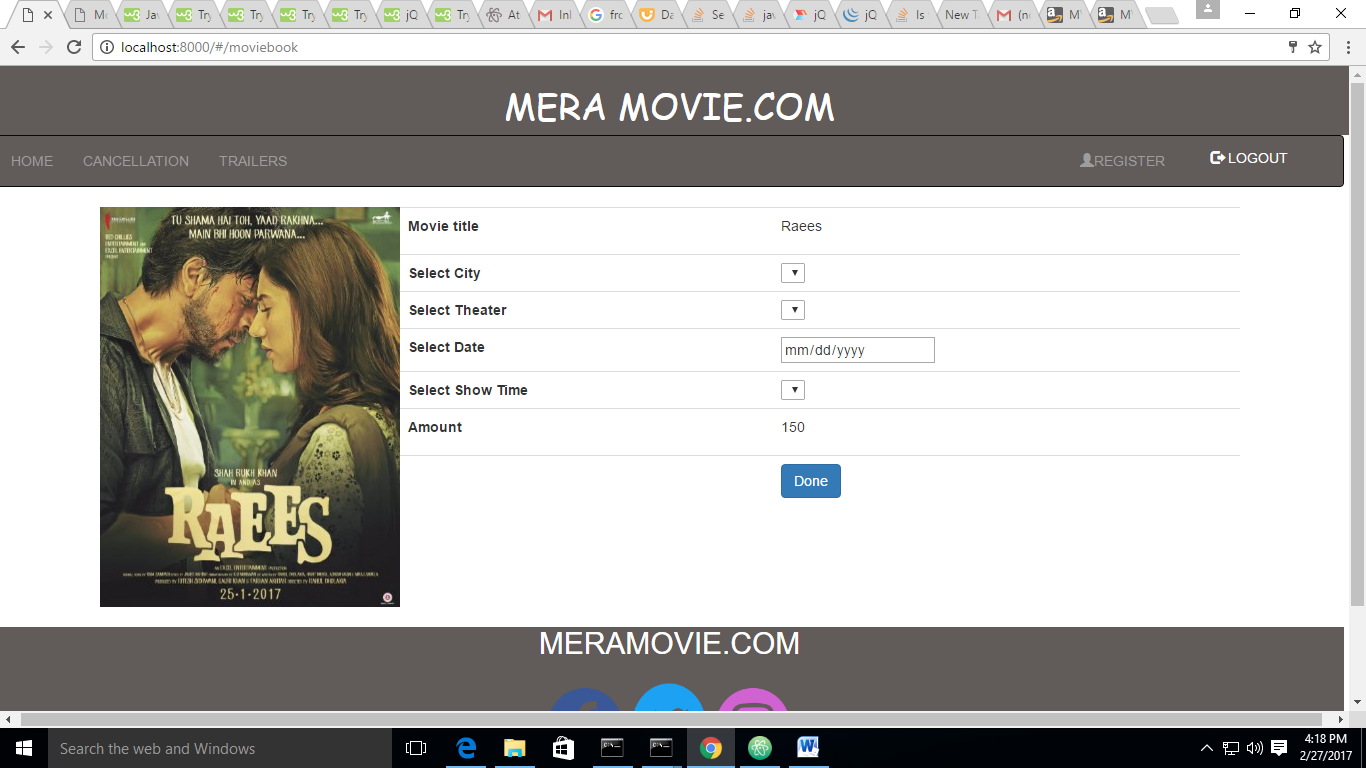
**Admin ShowTime assigning**

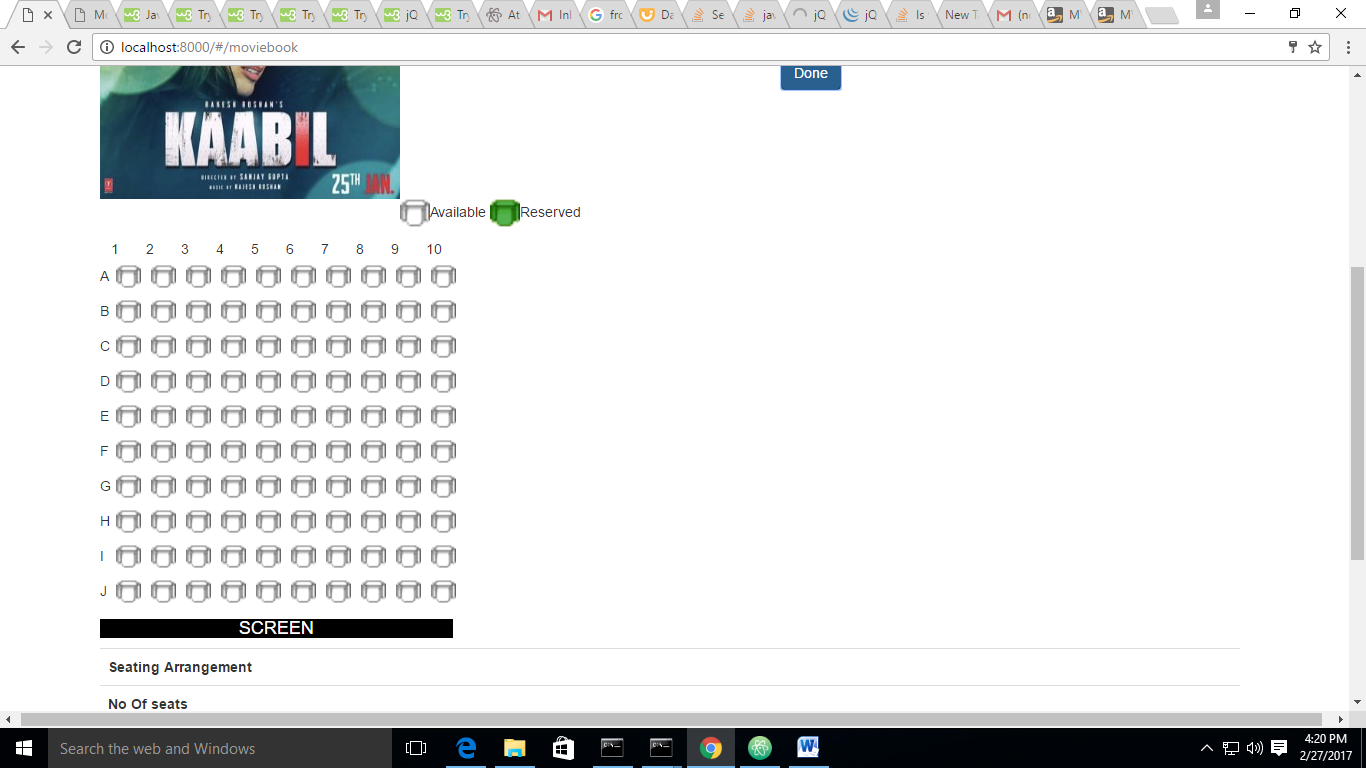


**Admin –assign cities**



**Booking**





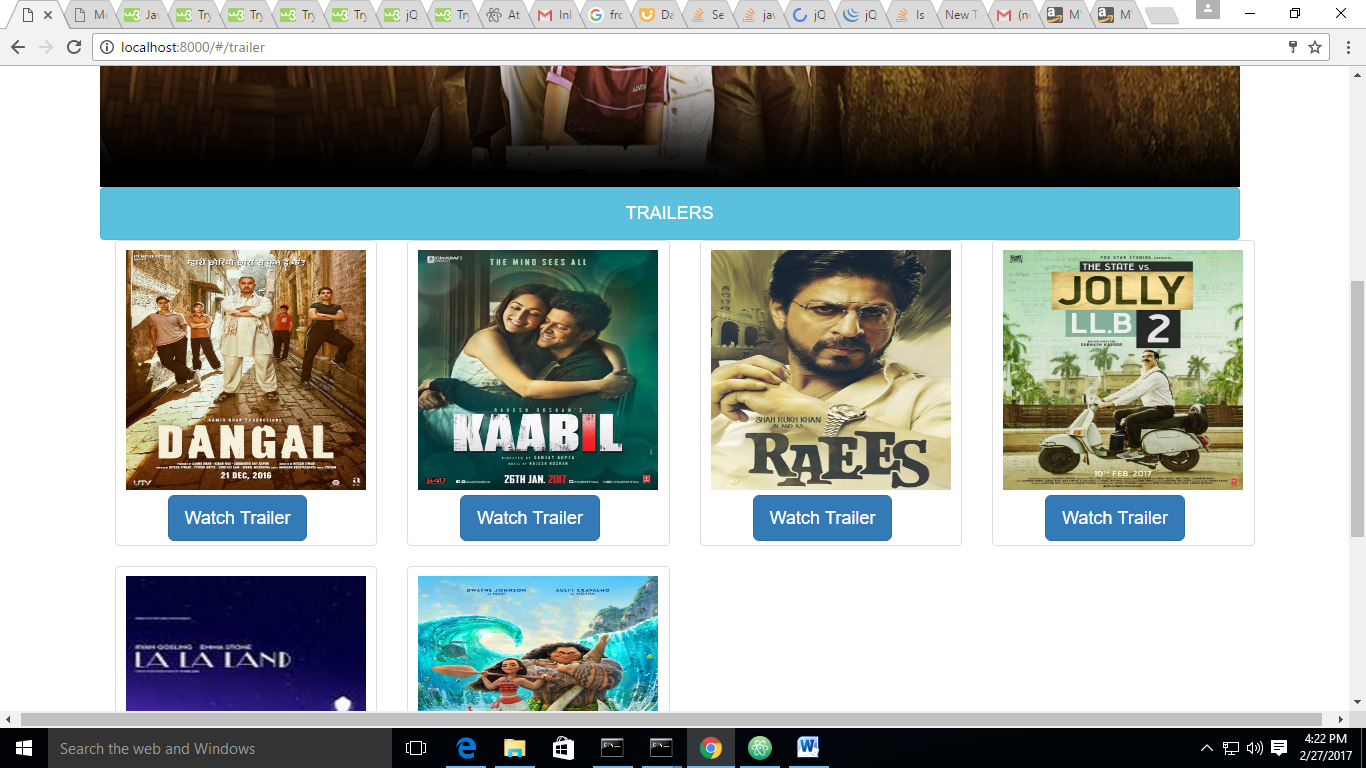
**Confirmation Page**



**Cancellation Page**



**Trailer Page**



**CHAPTER 8**

**REFERENCES**

**Website:**

* + 1. <http://www.w3schools.com/bootstrap/default.asp>
    2. <http://www.w3schools.com/css/default.asp>
    3. <http://www.w3schools.com/js/default.asp>
    4. <http://www.w3schools.com/angular.js/default.asp>