**NodeJS with MongoDb & Express-MEAN stack**

Lab Book

Document Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Revision No. | Author | **Summary of Changes** |
| April 2018 | 1.2 | Rahul Vikash | Created new lab book as per revised course contents |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

*Getting Started..……..…………………………………………………………………………… 4*

[NodeJS with MongoDb & Express 1](#_Toc514260345)

[Overview 4](#_Toc514260346)

[Getting Started 3](#_Toc514260347)

[Overview 3](#_Toc514260348)

[Setup Checklist for Node Js 3](#_Toc514260349)

[Instructions 3](#_Toc514260350)

[Lab 1.Module 5](#_Toc514260351)..............................................................................................................................5

[Lab 2.Mongo Db with Node JS 6](#_Toc514260352)

[Lab 3 Express Js with Mongo Db Node 7](#_Toc514260353)

Lab4 Express Js with PUG/JADE

Lab 5 Angular 5 with Mongo & Node Express ---MEAN Stack

[Appendix A: Table of Figures 9](#_Toc514260354)

Getting Started

## Overview

This lab book is a guided tour for learning Introduction to Object Orientedversion and above. It comprises ‘To Do’ assignments. Follow the steps provided to work out the ‘To Do’ assignments given.

## Setup Checklist for Node Js

Here’s what is expected on your machine for the lab in order to work.

Minimum System Requirements

* Intel Pentium 90 or higher (P166 recommended)
* Microsoft Windows XP, Windows 7 or Windows 8
* Memory: 2GB of RAM (1GB or more recommended)
* Google Chrome 36.0 or Mozilla Firefox 31.0 or Internet Explorer 10 or above
* Install Node JS

Please ensure that the following is done:

* A text editor like Notepad or Notepad++ or Eclipse Luna is installed.
* Visual studio Code

## Instructions

* Create a directory by your name in drive <drive>. In this directory, create a subdirectory JavaScript. For each lab exercise create a directory as lab <lab number>.

1. Module

|  |  |
| --- | --- |
| Goals | * Working on Node JS Modules & basics |
| Time | 60 minutes |
|  |  |

* 1. Create a **prob1.js** containing script. In this script, declare an array of 6 employee names and display it in the console using node .
  2. Create a node server read the data from the file and display on the console
  3. Create a node server write the data into the file
  4. Create a Web Application using node where user can enter username ,password & by using request & response data will transfer to next page

1. Mongo Db with Node JS

|  |  |
| --- | --- |
| Goals | * Working on Mongo Db & Node JS |
| Time | 120 minutes |

1. **Create a console base application using node & Mongo Db where we can do following operation .**
2. Insert the static Data such as product ID, product name, product cost & product description in Mongo Db using node
3. Get All Data
4. Get Date based on product id
5. Delete Data based on product id
6. Update data based on product id

Lab 3 Express Js with Mongo Db Node

|  |  |
| --- | --- |
| Goals | * Working on Node with Mongo |
| Time | 1. Minutes |

**3.1 Consider Below JSON File:**

var employees = {

employee: [

{

"empId": 1001,

"empName": "Jack",

"empSalary": 40000,

"empAddress": {

"city": "Pune",

"state": "Maharashtra"

}

},

{

"empId": 1002,

"empName": "Jill",

"empSalary": 42000,

"empAddress": {

"city": "Nashik",

"state": "Maharashtra"

}

},

{

"empId": 1003,

"empName": "Sebastian",

"empSalary": 14000,

"empAddress": {

"city": "Agra",

"state": "Uttar Pradesh"

}

},

{

"empId": 1004,

"empName": "Jessica",

"empSalary": 30000,

"empAddress": {

"city": "Lucknow",

"state": "Uttar Pradesh"

}

},

{

"empId": 1005,

"empName": "Terena",

"empSalary": 45000,

"empAddress": {

"city": "San Diego",

"state": "California"

}

},

{

"empId": 1006,

"empName": "Aadel",

"empSalary": 50000,

"empAddress": {

"city": "San Jose",

"state": "California"

}

},

]

}

Using Express TODO below retrievals

1. Displaying ALL employees

2. Displaying employees belonging to a specific state

3. Updating city of employee

4. Adding a new employee.

**3.2 Extend assignment 3.2 add all data in Mongo Db & do the following operation**

1. Displaying ALL employees

2. Displaying employees belonging to a specific state

3. Updating city of employee

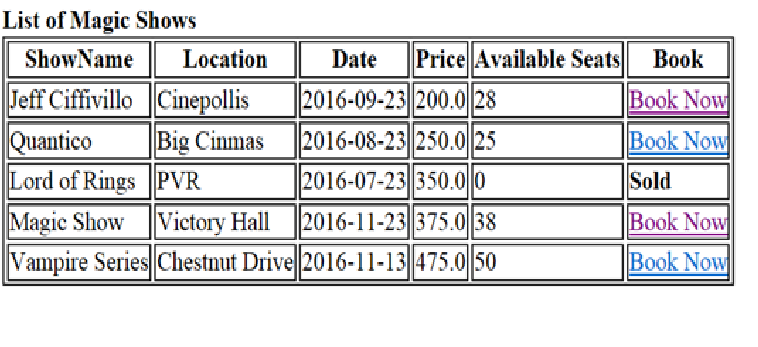
4. Adding a new employee.

Lab 4 Express Js with PUG/JADE

|  |  |
| --- | --- |
| Goals | * Working on PUG/JADE |
| Time | 1. Minutes |

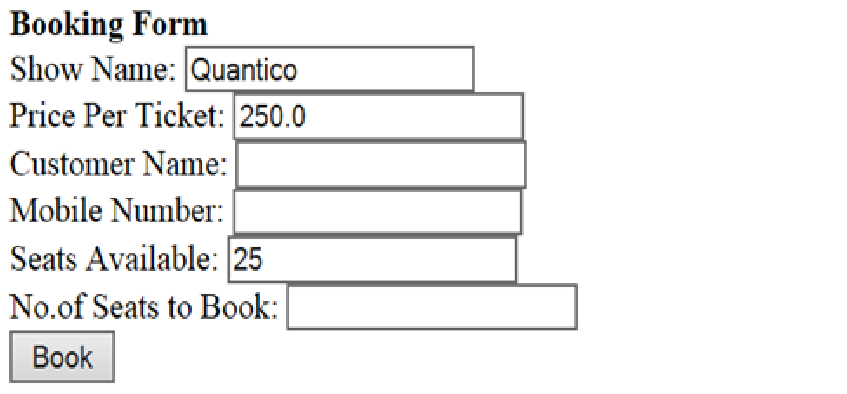
**Requirement:**MagicWorld is an online ticket booking system that enables users to book magic shows which are scheduled in different locations. A user can view the availability of seats in the show and book the show ticket. Payment module of this application is out of scope. Design the web application for the same using Node with Express using PUG/Jade.

Pug index.pug will show like this data is coming from JSON file



* If number of seats available for a particular show is greater than 0, then display “Book Now” link
* If number of seats available for a particular show is less than or equal to 0, then display “Sold” keyword.

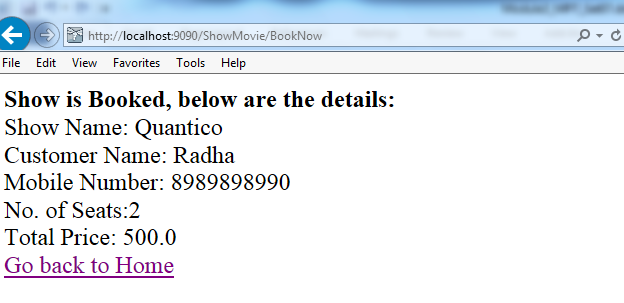
Once “BookNow” link is clicked, navigate to **bookNow.pug** page as shown below:



* ShowName, Price per Ticket and Seats Available needs to be populated
* After the Book button is clicked:

Update the seat column in JSON file value as (No of seats available –No of seats wants to book) in shows table for the particular show.

Open Success.pug with all value

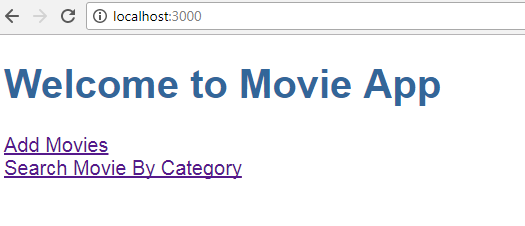


Lab 5: Angular 5 with Node, Express & Mongo DB—MEAN Stack

|  |  |
| --- | --- |
| Goals | * Angular5 with Node & Express JS & MongoDb |
| Time | 240 minutes |

Create a “Movie App” wherein movies could be maintained according to genres like Fiction, Drama and Satire. There is also a search option allowing to search for movies.

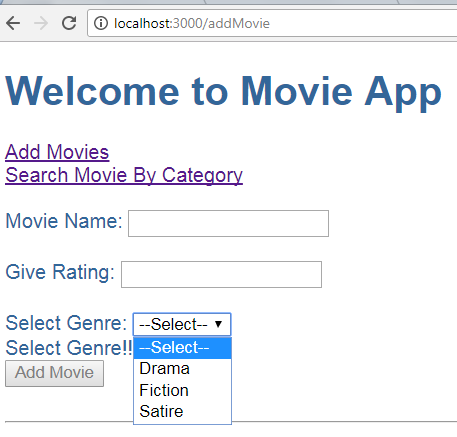
Refer below screen shot: Created in Angular 5



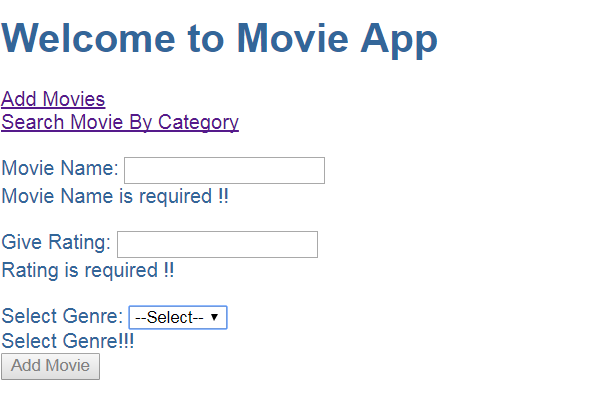
1. When the user clicks on “Add Movies” -> Allows a User to add a movie.

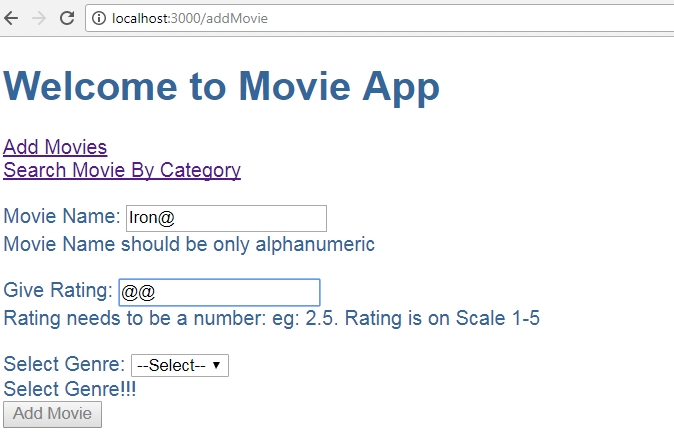
The user can add movie(s) depending on Genre (drop-down list loaded with values [Drama, Fiction, Satire])

The user could add multiple movies. On Adding Movies will be added in Mongo Db with the help of Node & Express (Restful-API). Use Node & Express RestFul-API with Mongo Db concept Refer below screen shot:



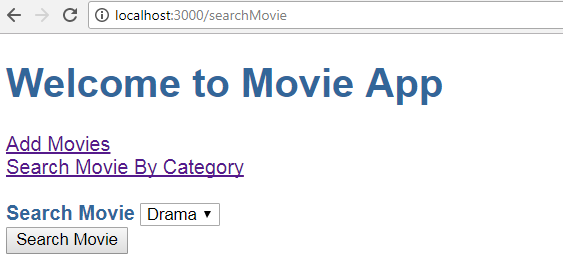
Refer below screens for validations:



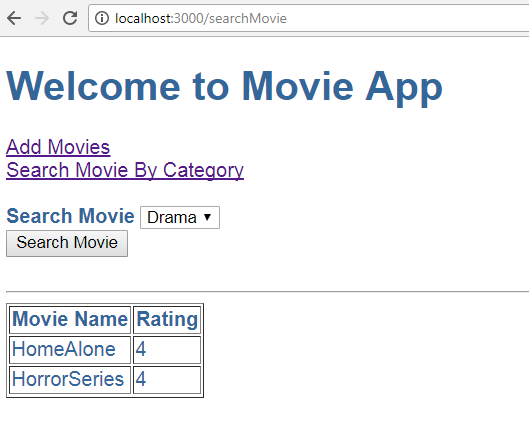


1. When the user clicks on “Search Movie By Category” -> Allows user to search movie(s) by Category. [Depending on Genre (Drama,Fiction,Satire)]

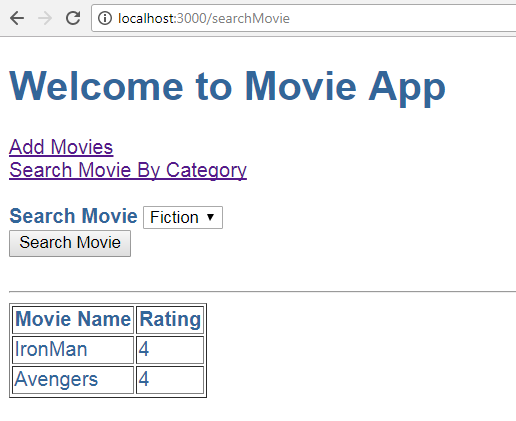
There could be more than one movies depending on same Genre. Refer below screen shots:



Thus if Searching Movie on ‘Drama’ displays below Result:



Thus if Searching Movie on ‘Fiction’ displays below Result:



**Note:**

1. In above assignment movies are added via “Add Movies” and Searched via “Search Movie By Category”. ***These are different components.***
2. Make use of Angular 5 Forms.
3. Make use of services.
4. Make use of Router Module. Which should be created as a separate module and included in the main App Module.
5. Use Node & Express restful API & Mongo DB concept to add movies & search movies.
6. All adding & searching movies done through MongoDb

## Appendix A: Table of Figures

[Figure 1 5](#_Toc452192526)

[Figure 2 5](#_Toc452192527)

[Figure 3 6](#_Toc452192528)

[Figure 4 6](#_Toc452192529)

[Figure 5 7](#_Toc452192530)

[Figure 6 8](#_Toc452192531)

[Figure 7 8](#_Toc452192532)

[Figure 8 8](#_Toc452192533)

[Figure 9 9](#_Toc452192534)

[Figure 10 10](#_Toc452192535)

[Figure 11 10](#_Toc452192536)