**Application Name:** Movie Booking system

**URL of the application:**

**About the application in a paragraph:** A simple movie booking built with MEAN stack which includes MongoDB, ExpressJS, AngularJS, NodeJS.

**Modules in the application:**

**Search Option:** Search a particular movie

**Filter Option:** Filters the movie based on genre and language.

**SignUP/SignIN :** User can Signin if already have an account or can create using Signup.

**Trailers:** Displays the latest video snippets.

**FAQ :** Displays the frequently asked questions based on booking, offer availability, etc

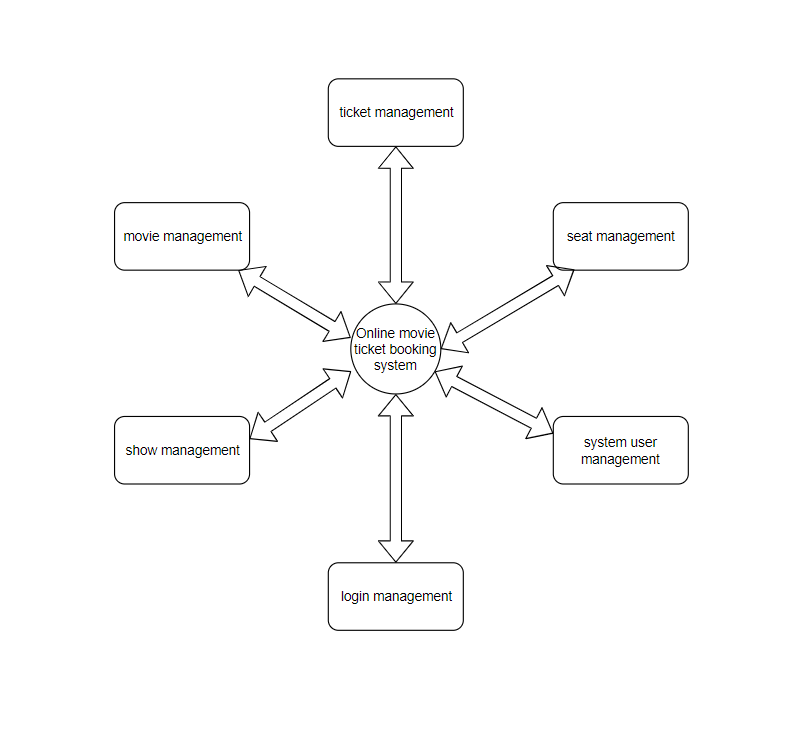
**Chat With Us :** This feature helps the customer to have a chat with customer care service.

**Offer :** Displays the offers available for a particular payment method.

**Book Tickets**: Book tickets by selecting date, time and seat availability.

**Payment** : The customers will be able to pay online and book the tickets for their respective show.

**Provide a work Break Down Diagram for the application:**



**Write in your own words how the application has used Mean Stack Development**

In my application, I have used the following tools:

MEAN

* MongoDB
  + As a document database to store and retrieve documents
  + Efficient filtering of data objects
* ExpressJs
  + A web framework to develop the backend
  + To develop a REST API to interface with any type of client
* Angular
  + As a frontend framework to develop the application UI
  + A component-based UI
* NodeJs
  + As a runtime for ExpressJS
  + Supports running of JavaScript on the server side

**List of technologies/languages/framework/library used with purpose and version**

|  |  |
| --- | --- |
| **Technologies/languages/framework/library** | **Purpose** |
| JavaScript | Full stack development (Frontend + backend) |
| MongoDB | NoSQL document database |
| ExpressJS | Web framework for the server application |
| NodeJS | A runtime environment for ExpressJS to run in the server |
| Angular | To develop the client-side application |
| PrimeNG | UI components for Angular |
| expressJWT | Authentication package using JWT |
| mongoose | NPM package to create data models and a simple Api fpr mongodb |
| dotenv | To load environment variables into the application |
| SCSS | A CSS compiler that provides a better way to write CSS |
| Typescript | To develop angular component modules |
| JSON | The standard medium of data transfer between server and client |

**Write in your own words how the following concepts have been used in your application:**

|  |  |
| --- | --- |
| HTML 5 | To develop basic UI components |
| CSS | To style the UI components |
| Angular | To develop the Client-side application |
| Node JS | As a runtime for JavaScript to run at server side |
| Express JS | As a web framework to develop the server-side application and the REST API |
| JSON | As a standard data transfer medium between the server and client |
| MongoDB | A database for storing and retrieving JSON documents |

|  |  |
| --- | --- |
| Number of Forms | 4 |
| Form Names for Evaluating the concepts of AngularJS,NodeJS,ExpressJS | SignUP/SignIN, FAQ, Chat With Us,payment |
| Tables for Evaluating the MongoDB concepts | SignUP/SignIN, FAQ, Chat With Us,payment |

For the following tables refer the concepts provided for learning from internet resources and fill the table

Weightage : 5 Marks

|  |  |  |
| --- | --- | --- |
| **Concept used from HTML5** | File Name | Additional Information |
| Form controls | \* | Implemented check boxes, text, numbers, email, |
| Semantic tags | \* | Main, header, center, footer etc |
| Table | \* | Table to display data |

Weightage : 10 Marks

|  |  |  |
| --- | --- | --- |
| **Concept used from XML** | File Name | Additional Information |
| Xml | Xml, xsl | Done the implementation using xml |

Weightage : 10 Marks

|  |  |  |
| --- | --- | --- |
| **Concept used from JavaScript** | File Name | Additional Information |
| Const, let, var | \* | Used to create constants and variables |
| Arrays | Models/\* | Used to store multiple objects of database information |
| Objects | App.js | Used the response and request objects to extract information |
| Template strings | App.js | Use to embed variables dynamically in a string |
| CommonJS Imports | \* | To import and export modules |
| Arrow functions | \* | Used in place of functions wherever possible |
| **Concept used from JSON** | **File Name** | **Additional Information** |

|  |  |  |
| --- | --- | --- |
| Array | APP.JS | To provide cration information to the angular app |
| Object | APP.JS | Config information to angular app |
| Array of objects | APP.JS | To provide config info about multiple objects |

Weightage : 20 Marks

|  |  |  |
| --- | --- | --- |
| **Concepts used from Angular** | File Name | Additional Information |
| Angular html components | FORMS | HTML components that will be rendered on the browser |
| Angular data models | FORMS | Define the format of data that will be fetched from the server |
| Angular services | FORMS | Defines logic that will fetch data and interact with the REST API |

Weightage : 20 Marks

|  |  |  |
| --- | --- | --- |
| **Concepts used from ExpressJS** | File Name | Additional Information |
| Routing | App.js, routes/ | Routing requests for various API endpoints |
| Mongoose Models | models/ | Creating data models to store in DB |
| Environment Variables | .env | Load up ENV variables into the application |
| Middleware | \*.js | JSON parser, error handler, authentication middleware, static file storage |

Weightage : 20 Marks

|  |  |  |
| --- | --- | --- |
| **Concepts used from NodeJS** | File Name | Additional Information |
| dotenv | .env | To load ENV files into the application |
| file | App.js | Used nodejs file handling library |
|  |  |  |
| path | App.js | Used nodejs path parsing and handling libraries |
| Request and response | App.js | Used the req and res objects to handle requests and responses |

Weightage : 10 Marks

|  |  |  |
| --- | --- | --- |
| **Concepts used from MongoDB** | File Name | Additional Information |
| Data Model | App.js | Using mongoose to create models of each entities in the database |
| Fetch | `` | Fetch documents from the DB |
| Edit | `` | Update documents in the DB |
| Delete | `` | Delete documents from the DB |
| Filter | `` | Filter documents based on the query |
| MongoDB Atlas | App.js | Used mongoose to connect to MongoDB’s cloud database |

## Provide five features that can be added to application in future:

1. Payment through the system
2. Notifications
3. Schema Updates
4. UI Modifications