

## **# ZomentumTask**

### **Movie Theatre Ticket Booking API services**

#### **#Tech and Frameworks used:**

For Database: MongoDB

Runtime Environment: NodeJs

Framework: ExpressJs

#### **#Functionalities involved:**

- 1.Book a Movie Ticket.
- 2.Update the previously booked Ticket Timings.
- 3.View all the tickets for a particular time/show.
- 4.Delete a particular ticket.
- 5.View User based on ticket ID.

#### **#Schema Involved**

As we gonna have shows for which user books a ticket.So,a Show Schema is there which contain some information about show.

##### **Show Schema:**

StartTime: When the Show started,

EndTime: When the show end,

Date: Date of the show,

SeatAvailable: Number of seats which are free / to be filled

StartTime, EndTime & Date can together point out a show with information of number of seats available.

For this, I have made some dataset which is as:

ZomentumTask.shows

DOCUMENTS 8 TOTAL SIZE 1018.0B AVG. SIZE 127B INDEXES 1 TOP 3

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER OPTIONS FIND

ADD DATA VIEW

Displaying documents 1 - 8 of 8

```

{
  "_id": "5f4a9b06a504f976563fb71d",
  "startTime": "15:00:00",
  "endTime": "18:00:00",
  "date": "01-09-2020",
  "seatAvailable": "20"
}

{
  "_id": "5f4a9b2ca47763d24bbf936",
  "startTime": "12:00:00",
  "endTime": "15:00:00",
  "date": "01-09-2020",
  "seatAvailable": "20"
}

{
  "_id": "5f4a9b42a47763d24bbf937",
  "startTime": "6:00:00",
  "endTime": "9:00:00",
  "date": "02-09-2020",
  "seatAvailable": "20"
}

{
  "_id": "5f4a9b55a47763d24bbf938",
  "startTime": "9:00:00",
  "endTime": "12:00:00",
  "date": "02-09-2020",
  "seatAvailable": "20"
}

{
  "_id": "5f4a9b5ea47763d24bbf939",
  "startTime": "15:00:00",
  "endTime": "18:00:00",
  "date": "03-09-2020"
}

```

After that, there is a Ticket Schema involved which contains following information,

#### Ticket Schema:

Ticket Id: Uniquely generated at time of ticket booking,

Booking Timing: Time when ticket was booked,

UserDetails:{

    username: Name of user booking the ticket,

    phone: Mobile Number of the user

}

ShowReference: ObjectId of the show of which User is booking the ticket

Key-Value representation-

```

{
  "_id": ObjectId("5f4b22029d5a824138b0292c"),
  "userDetails": {
    "username": "piyush",
    "phone": 7248578044,
    "bookingTime": 2020-08-30T03:50:26.247+00:00,
    "showRef": ObjectId("5f4a9b06a504f976563fb71d"),
    "ticketId": "ID-z09idcvsegk30ei"
  },
  "__v": 0
}

```

#### #Book a Ticket

API will get some information and a command to book a ticket.

Information received will be:

```
{
  "username" : Name of the User,
  "phone" : Phone Number of the user,
  "date" : Date of the show,
  "startTiming" : - ,
  "endTiming" : -
}
```

After that program will validate if format of information received is valid or not. If not, it will send a error message to user.

POST http://localhost:3000/book

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "username" : "piyush",
3   "phone" : "72485704",
4   "date" : "29-15-2020",
5   "startTiming" : "25:00:00",
6   "endTiming" : "188:00:00"
7 }
```

Body Cookies Headers (6) Test Results

Pretty Raw Preview Visualize HTML

```
1 **Date Format is not Valid
2 **Phone Number is not Valid
3 **Start Time is not Valid
4 **End Time is not Valid
```

If all the information received is valid, then program check if such show timing is available or not. If not then:

POST http://localhost:3000/book

Params Authorization Headers (8) **Body** Pre-request

● none ● form-data ● x-www-form-urlencoded ● **raw** ● |

```
1 {
2   "username" : "piyush",
3   "phone" : "7248578704",
4   "date" : "29-08-2020",
5   "startTiming" : "13:00:00",
6   "endTiming" : "15:00:00"
7 }
```

Body Cookies Headers (6) Test Results

Pretty Raw Preview Visualize HTML ↕

1 Show not Available

**\*\*Show Not Available\*\***

If all OK, then program book a ticket for that show and after that **decrease a seatAvailable value in that show by One(1)**.

Output user get:

POST http://localhost:3000/book

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

● none ● form-data ● x-www-form-urlencoded ● **raw** ● binary ● GraphQL **JSON** ▼

```
1 {
2   "username" : "RandomUser",
3   "phone" : "9876543210",
4   "date" : "02-09-2020",
5   "startTiming" : "9:00:00",
6   "endTiming" : "12:00:00"
7 }
```

body Cookies Headers (6) Test Results

Pretty Raw Preview Visualize JSON ▼

```
1 {
2   "status": "Ticket Booked Sucessfully!! Store your Ticket ID for later use..",
3   "ticketID": "ID-z09idhj0kegx7zy5",
4   "username": "RandomUser",
5   "phone": 9876543210,
6   "bookingTime": "2020-08-30T09:58:13.946Z",
7   "showStartTime": "9:00:00",
8   "showEndTime": "12:00:00",
9   "DateofShow": "02-09-2020"
10 }
```

**Ticket Id is uniquely generated using uniqid module.**

Ticket saved in Database:

## Zomentum lask.tickets

Documents

Aggregations

Schema

Expl

FILTER

ADD DATA

VIEW

```
  _id: ObjectId("5f4b7835d7e9fd58bc650e99")
  ✓ userDetails: Object
    username: "RandomUser"
    phone: 9876543210
    bookingTime: 2020-08-30T09:58:13.946+00:00
    showRef: ObjectId("5f4a9b55a477763d24bbf938")
    ticketId: "ID-z09idhj0kegx7zy5"
    __v: 0
```

If seats not available:

```
  _id: "5f4a9b55a477763d24bbf938"
  startTime: "9:00:00"
  endTime: "12:00:00"
  date: "02-09-2020"
  seatAvailable: 0
```

Output:

POST http://localhost:3000/book

Params Authorization Headers (8) **Body** Pre-request

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw

```
1 {
2   "username" : "piyush",
3   "phone" : "7248578589",
4   "date" : "02-09-2020",
5   "startTiming" : "9:00:00",
6   "endTiming" : "12:00:00"
7 }
```

Body Cookies Headers (6) Test Results

Pretty Raw Preview Visualize HTML

```
1 Housefull !! Try for another show
```

HouseFull!!!!!!

## #Update Ticket Timings

Information received by the program:

```
{
  "ticketId" : TicketId for which timing will beupdated,
  "newDate" : Updated Date,
  "newStartTiming": Updated start time,
  "newEndTiming": Updated End Time
}
```

Validation will done for all this information. After that, program will update the seatAvailable in previous show timings(i.e; increase it by 1).

And then check for the seat availability in new show timings.

If all OK, then it will update the timings.

POST http://localhost:3000/update

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

● none ● form-data ● x-www-form-urlencoded ● **raw** ● binary ● GraphQL **JSON** ▼

```
1 {
2   "ticketId" : "ID-z09idhj0kegx7zy5",
3   "newDate" : "02-09-2020",
4   "newStartTiming": "12:00:00",
5   "newEndTiming": "15:00:00"
6 }
7
```

Body Cookies Headers (6) Test Results

Pretty Raw Preview Visualize JSON ▼ ↺

```
1 {
2   "status": "Ticket Updated Sucessfully!! Store your Ticket ID for later use..",
3   "ticketID": "ID-z09idhj0kegx7zy5",
4   "username": "RandomUser",
5   "phone": 9876543210,
6   "updateTime": "2020-08-30T09:58:13.946Z",
7   "updatedshowStartTime": "12:00:00",
8   "updatedShowEndTime": "15:00:00",
9   "updatedDateofShow": "02-09-2020"
10 }
```

Show Reference Changed:

```
> {
  _id: ObjectId("5f4b7835d7e9fd58bc650e99")
  userDetails: Object
    username: "RandomUser"
    phone: 9876543210
    bookingTime: 2020-08-30T09:58:13.946+00:00
    showRef: ObjectId("5f4a9b66a477763d24bbf93a")
    ticketId: "ID-z09idhj0kegx7zy5"
    __v: 0
}
```

Number of Available Seats Decreased for that show:



```
_id: "5f4a9b5ea477763d24bbf939"  
startTime: "15:00:00"  
endTime: "18:00:00"  
date: "02-09-2020"  
seatAvailable: "20"
```

```
_id: "5f4a9b66a477763d24bbf93a"  
startTime: "12:00:00"  
endTime: "15:00:00"  
date: "02-09-2020"  
seatAvailable: 19
```

## #Deleting the ticket

Program simple remove the ticket from the database corresponding to TicketId received from client.

Output:

The screenshot shows a REST client interface with the following components:

- Request Section:**
  - Method: `DELETE`
  - URL: `http://localhost:3000/ticket/ID-z09idhj0kascx859`
  - Body: `{ "ticketId": "ID-z09idhj0kascx859" }`
- Response Section:**
  - Status: `200 OK`
  - Body: `**Ticket Deleted Success!!**`

Before Deleting:

## ZomentumTask.tickets

The screenshot shows the MongoDB Compass interface for the 'ZomentumTask.tickets' collection. The 'Documents' tab is active. A toolbar includes a 'FILTER' button, an 'ADD DATA' button, and icons for uploading, viewing, and switching between list, JSON, and grid views. Two documents are displayed in the list view.

```
{
  "_id": ObjectId("5f4b7835d7e9fd58bc650e99"),
  "userDetails": {
    "username": "RandomUser",
    "phone": 9876543210,
    "bookingTime": "2020-08-30T09:58:13.946+00:00",
    "showRef": ObjectId("5f4a9b66a477763d24bbf93a"),
    "ticketId": "ID-z09idhj0kegx7zy5"
  },
  "__v": 0
}
```

```
{
  "userDetails": {
    "username": "RandomUser2",
    "phone": 9876576210,
    "bookingTime": "2020-08-30T09:58:13.946+00:00",
    "showRef": ObjectId("5f4a9b66a477763d24bbf93a"),
    "ticketId": "ID-z09idhj0kascx859"
  },
  "__v": 0,
  "_id": ObjectId("5f4b7f34a477763d24bbf93d")
}
```

After Deleting:

# ZomentumTask.tickets

Documents

Aggregations

Schema

FILTER

ADD DATA

VIEW

```
_id: ObjectId("5f4b7835d7e9fd58bc650e99")
> userDetails: Object
  bookingTime: 2020-08-30T09:58:13.946+00:00
  showRef: ObjectId("5f4a9b66a477763d24bbf93a")
  ticketId: "ID-z09idhj0kegx7zy5"
  __v: 0
```

## #View the tickets for particular timings

Information received from client Side will be:

```
{
  "date" : "02-09-2020",
  "startTiming" : "12:00:00",
  "endTiming" : "15:00:00"
}
```

Values are there for demo purpose only.

Validation of timing will processed, if all OK, program will fetch all the ricket details from the database for that particular timing and return to client side.

Entries in Database:

# ZomentumTask.tickets

Documents

Aggregations

Schema

Explain Plan

FILTER

ADD DATA

VIEW

>

\_id: ObjectId("5f4b80c9d95ab93fbc801a0f")

userDetails: Object

username: "piyush"

phone: 9875643268

bookingTime: 2020-08-30T10:34:49.018+00:00

showRef: ObjectId("5f4a9b66a477763d24bbf93a")

ticketId: "ID-z09idcl8kegyj1od"

\_\_v: 0

>

\_id: ObjectId("5f4b80dbd95ab93fbc801a10")

userDetails: Object

username: "User2"

phone: 9875634561

bookingTime: 2020-08-30T10:35:07.033+00:00

showRef: ObjectId("5f4a9b66a477763d24bbf93a")

ticketId: "ID-z09idcl8kegyjfkp"

\_\_v: 0

>

\_id: ObjectId("5f4b80eed95ab93fbc801a11")

userDetails: Object

username: "User3"

phone: 9524796761

bookingTime: 2020-08-30T10:35:26.389+00:00

showRef: ObjectId("5f4a9b66a477763d24bbf93a")

ticketId: "ID-z09idcl8kegyjujd"

\_\_v: 0

>

\_id: ObjectId("5f4b80f6d95ab93fbc801a12")

userDetails: Object

username: "User4"

phone: 9524795661

Response:

POST

http://localhost:3000/viewTickets

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

```
1  {
2    "date": "02-09-2020",
3    "startTiming": "12:00:00",
4    "endTiming": "15:00:00"
5  }
```

Body

Cookies

Headers (6)

Test Results

Pretty

Raw

Preview

Visualize

JSON

```
1  [
2    {
3      "status": "Ticket Booked Sucessfully!! Store your Ticket ID for later use..",
4      "ticketID": "ID-z09idhj0kegx7zy5",
5      "username": "RandomUser",
6      "phone": 9876543210,
7      "bookingTime": "2020-08-30T09:58:13.946Z",
8      "showStartTime": "12:00:00",
9      "showEndTime": "15:00:00",
10     "DateofShow": "02-09-2020"
11   },
12   {
13     "status": "Ticket Booked Sucessfully!! Store your Ticket ID for later use..",
14     "ticketID": "ID-z09idc18kegyjlod",
15     "username": "piyush",
16     "phone": 9875643268,
17     "bookingTime": "2020-08-30T10:34:49.018Z",
18     "showStartTime": "12:00:00",
19     "showEndTime": "15:00:00",
20     "DateofShow": "02-09-2020"
21   },
22   {
23     "status": "Ticket Booked Sucessfully!! Store your Ticket ID for later use..",
24     "ticketID": "ID-z09idc18kegyjfkp",
25     "username": "User2",
26     "phone": 9875634561,
27     "bookingTime": "2020-08-30T10:35:07.033Z",
28     "showStartTime": "12:00:00",
29     "showEndTime": "15:00:00",
30     "DateofShow": "02-09-2020"
31   },
22   {
23     "status": "Ticket Booked Sucessfully!! Store your Ticket ID for later use..",
24     "ticketID": "ID-z09idc18kegyjfkp",
25     "username": "User2",
26     "phone": 9875634561,
27     "bookingTime": "2020-08-30T10:35:07.033Z",
28     "showStartTime": "12:00:00",
29     "showEndTime": "15:00:00",
30     "DateofShow": "02-09-2020"
31   },
```

```

30         DateofShow : 02-09-2020
31     },
32     {
33         "status": "Ticket Booked Sucessfully!! Store your Ticket ID for later use..",
34         "ticketID": "ID-z09idcl8kegyjuid",
35         "username": "User3",
36         "phone": 9524796761,
37         "bookingTime": "2020-08-30T10:35:26.389Z",
38         "showStartTime": "12:00:00",
39         "showEndTime": "15:00:00",
40         "DateofShow": "02-09-2020"
41     },
42     {
43         "status": "Ticket Booked Sucessfully!! Store your Ticket ID for later use..",
44         "ticketID": "ID-z09idcl8kegyk111",
45         "username": "User4",
46         "phone": 9524795661,
47         "bookingTime": "2020-08-30T10:35:34.857Z",
48         "showStartTime": "12:00:00",
49         "showEndTime": "15:00:00",
50         "DateofShow": "02-09-2020"
51     }
52 ]

```

### #View the User by using TicketId

Ticket Id will be received from the client side and then output will be:

POST

http://localhost:3000/viewUser

Params

Authorization

Headers (8)

Body ●

Pre-request Script

Tests

Settings

☐ none

☐ form-data

☐ x-www-form-urlencoded

☒ raw

☐ binary

☐ GraphQL

JSON ▼

1 {

2     "ticketId" : "ID-z09idcl8kegyj1od"

3 }

Body

Cookies

Headers (6)

Test Results

Pretty

Raw

Preview

Visualize

JSON ▼

1 {

2     "username": "piyush",

3     "phone": 9875643268

4 }