# **Build the theater** resource APIs



Relevel

by Unacademy

#### **Agenda**

- Setting up the model for the Theatre resource
- Ability to create a new Theatre
- Ability to update an existing Theatre
- Ability to get all the Theatres
- Ability to filter Theatre based on the pin/city
- Ability to delete an existing Theatre

#### **Data Models**

- Theatre Resource
- Name It should be string and required.
- Description It should be string and required.
- City It should be string and required.
- Pin Code It should be string and required.
- Created At It should be Date and immutable with default value.
- Updated At It should be Date with default value.

# **Theatre Resource Application - CRUD Operations**

Create all the routes that will be required [Code Link] - https://github.com/Vishwa07dev/mba\_backend/blob/session2/routes/theatre.routes.js

```
createTheatre - "/mba/api/v1/theatres" updateTheatre - "/mba/api/v1/theatres/:id" getAllTheatres - "/mba/api/v1/theatres" getTheatre - "/mba/api/v1/theatres/:id" deleteTheatre - "/mba/api/v1/theatres/:id"
```

```
const theatreController = require("../controllers/theatre.controller");
module.exports = function (app) {
    app.get("/mba/api/v1/theatres", theatreController.getAllTheatres);
    app.get("/mba/api/v1/theatres/:id", theatreController.getTheatre);
    app.post("/mba/api/v1/theatres", theatreController.createTheatre);
    app.put("/mba/api/v1/theatres/:id", theatreController.updateTheatre);
    app.delete("/mba/api/v1/theatres/:id", theatreController.deleteTheatre);
}
```

#### **Create a New Theatre**

In this API, we are creating a theatre.

[Code Link] - https://github.com/Vishwa07dev/mba\_backend/blob/session2/controllers/theatre.controller.js

- 1. We are fetching the reg body data and create the object which will be used to store data in db
- 2. We are creating the theatre entry in the db and waiting for the promise using await.
- 3. On successful creation of Theatre we are responding back with success code 201

Here, we will use the Theatre schema of our database and create a theatre by giving all the details as an object to create the function of the Theatre

We will give below details to create a Theatre -

- Theatre Name
- Description
- City
- pinCode



```
exports.createTheatre = async (req, res) => {
    const theatreObject = {
        name: req.body.name,
        city: req.body.city,
        description: req.body.description,
        pinCode: req.body.pinCode
    }
    const theatre = await Theatre.create(theatreObject);
    res.status(201).send(theatre);
}
```

# Request

```
POST
         http://localhost:8080/mba/api/v1/theatres
                                                                                                                                 Send
      Authorization Headers (8)
                          Body Pre-request Script
                                                   Settings
                                                                                                                                    Cookies
Beautify
 1
     ... "name": "testThreater",
     ···"city": "Delhi",
     "description": "Very Spacious",
 5
       -"pinCode": 110055
  6
```

#### Update the theatre based on the theatre id

In this API, we will update a theatre based on the theatre id

#### [Code Link] -

https://github.com/Vishwa07dev/mba\_backend/blob/session2/controllers/theatre.controller.js

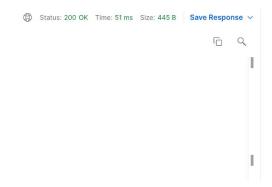
- Here, first, we will fetch the theatre based on the theatre id given in the request. We will use the findOne function of the theatre schema to fetch the theatre.
- If the theatre does not exist, we will send an error message saying that Theatre being updated doesn't exist"
- If the theatre exists, we will proceed with updating the theatre
- We will fetch all the details of the theatre from the request one by one.
- Once done, we will use the save function of the theatre schema to save the updated theatre in the database.



```
exports.updateTheatre = async (reg, res) => {
   const savedTheatre = await Theatre.findOne({ id: req.params.id });
   if (!savedTheatre) {
       return res.status(400).send({
           message: "Theatre being updated doesn't exist"
      });
   savedTheatre.name = req.body.name != undefined ? req.body.name :
savedTheatre.name:
   savedTheatre.description = req.body.description != undefined ?
req.body.description : savedTheatre.description;
   savedTheatre.city = req.body.city != undefined ? req.body.city :
savedTheatre.city;
   savedTheatre.pinCode = req.body.pinCode != undefined ? req.body.pinCode :
savedTheatre.pinCode;
   var updatedTheatre = await savedTheatre.save();
   res.status(200).send(updatedTheatre);
```

#### **Request**

```
| Pretty | Raw | Preview | Visualize | JSON | Fretty | Frett
```



#### **Get the Theatre**

In this API, we will get a list of the theatres.

[Code Link] -

https://github.com/Vishwa07dev/mba\_backend/blob/session2/controllers/theatre.controller.js

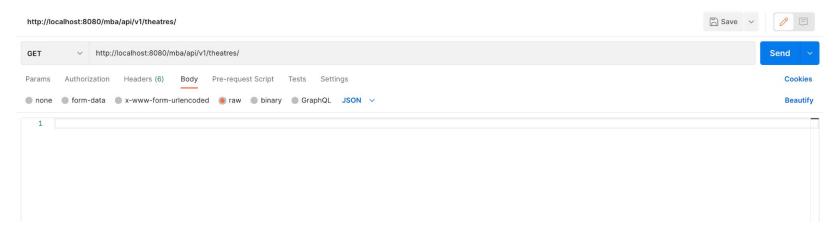
Here, we will use the find function of the Theatre schema present in our database.

- We also have used a query object here for the Theatre name or Theatre city or Theatre pincode. If the Theatre will be provided then it will give a Theatre based on the query itself
- If we are not giving a name, city and pincode in the API as a query parameter, it will give a list of all the theatre



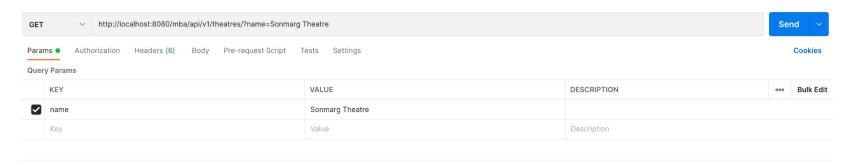
```
exports.getAllTheatres = async (req, res) => {
  const queryObj = {};
  if (req.query.name != undefined) {
      queryObj.name = req.query.name;
  if (req.query.city != undefined) {
      queryObj.city = req.query.city;
  if (req.query.pinCode != undefined) {
      queryObj.pinCode = req.query.pinCode;
  const theatres = await Theatre.find(queryObj);
  res.status(200).send(theatres);
```

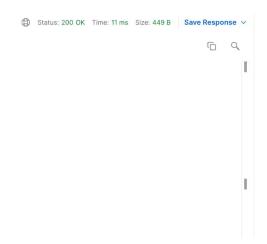
# Request – Get All the theatre



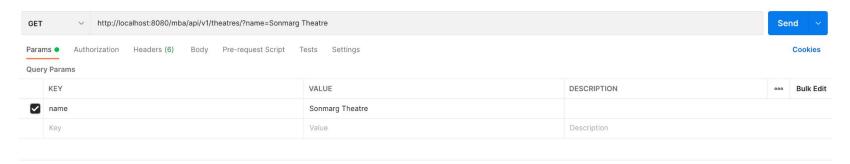
```
Body Cookies Headers (7) Test Results
                                                                                                                                   Status: 200 OK Time: 27 ms Size: 1.69 KB Save Response V
                                                                                                                                                                                [ Q
  Pretty
                  Preview Visualize JSON V
   2
               "_id": "624ce4fe607df2cd6a27cdc8",
               "name": "FunCinemas",
               "description": "Top class theatre",
               "city": "Bangalore",
               "pinCode": 560052,
               "createdAt": "2022-04-06T00:55:26.188Z",
               "updatedAt": "2022-04-06T00:55:26.188Z",
   10
               " v": 0
   11
   12
  13
               " id": "624ce4fe607df2cd6a27cdca",
  14
               "name": "PVR Cinemas - Kormangala",
   15
               "description": "PVR franchise theatre",
   16
               "city": "Bangalore",
  17
               "pinCode": 560095,
   18
               "createdAt": "2022-04-06T00:55:26.275Z",
  19
               "updatedAt": "2022-04-06T00:55:26.275Z",
   20
               "__v": 0
  21
   22
   23
                "_id": "624ce4fe607df2cd6a27cdcc",
   24
               "name": "IMax",
   25
               "description": "IMax franchise theatre",
   26
               "city": "Bangalore",
   27
               "pinCode": 560095,
   28
               "createdAt": "2022-04-06T00:55:26.277Z",
   29
               "updatedAt": "2022-04-06T00:55:26.277Z",
   30
               "__v": 0
  31
   32
   33
               "_id": "624ce4fe607df2cd6a27cdce",
   34
               "name": "Vaibhav Theatre",
   35
               "description": "Economical theatre",
   36
               "city": "Bangalore",
```

#### Request – Get the theatre based on the theatre name





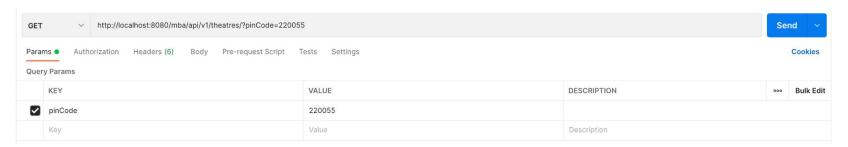
# Request – Get the theatre based on the theatre city



```
Body Cookies Headers (7) Test Results
  Pretty
                   Preview
                              Visualize
               " id": "624ce4fe607df2cd6a27cdc8",
                "name": "FunCinemas",
               "description": "Top class theatre",
               "city": "Bangalore",
               "pinCode": 560052,
               "createdAt": "2022-04-06T00:55:26.188Z",
               "updatedAt": "2022-04-06T00:55:26.188Z",
   10
               " v": 0
   11
   12
   13
               "_id": "624ce4fe607df2cd6a27cdca",
               "name": "PVR Cinemas - Kormangala",
   14
               "description": "PVR franchise theatre",
   15
   16
               "city": "Bangalore",
   17
               "pinCode": 560095.
   18
               "createdAt": "2022-04-06T00:55:26.275Z",
   19
               "updatedAt": "2022-04-06T00:55:26.275Z",
   20
               "__v": 0
   21
   22
   23
               "_id": "624ce4fe607df2cd6a27cdcc",
   24
               "name": "IMax",
   25
               "description": "IMax franchise theatre",
   26
               "city": "Bangalore",
   27
                "pinCode": 560095,
```

```
Status: 200 OK Time: 14 ms Size: 1.08 KB Save Response V
```

# Request – Get the theatre based on the theatre pincode





#### Get the theatre based on the theatre id

In this API, we will get a theatre based on the theatre id

[Code Link] -

https://github.com/Vishwa07dev/mba\_backend/blob/session2/controllers/theatre.c ontroller.js

Here, we will use the findOne function of the theatre schema present in our database.

We will pass theatre id as a parameter to the findOne function of the theatre schema. This will give us the theatre having theatre id equals to the passed one

```
exports.getTheatre = async (req, res) => {
    const theatre = await Theatre.findOne({
        _id: req.params.id
    });
    res.status(200).send(theatre);
}
```

# Request



```
| Pretty | Raw | Preview | Visualize | JSON | Visualize | JSON | Pretty | Raw | Preview | Previe
```



#### Delete the theatre based on the theatre id

In this API, we will delete a theatre based on the theatre id

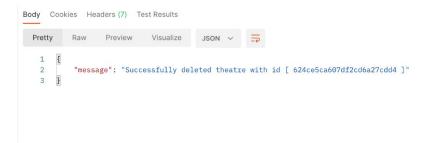
[Code Link] - https://github.com/Vishwa07dev/mba\_backend/blob/session2/controllers/theatre.controlle r.js

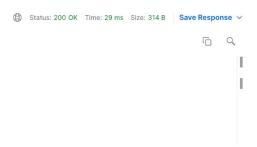
Here, we will use the deleteOne function of the theatre schema present in our database. We will pass the theatre id as a parameter to the deleteOne function of the theatre schema. This will delete the theatre having theatre id equals to the passed one

```
exports.deleteTheatre = async (req, res) => {
    await Theatre.deleteOne({
        _id: req.params.id
    });
    res.status(200).send({
        message: "Successfully deleted theatre with id [ " + req.params.id + " ]"
    });
}
```

# Request







#### **MCQ**

- 1. Which function of mongoDB Schema is used to find a single document?
- A. findOne()
- B. find()
- C. Both A and B
- D. None

**Answer: A** 

- 2. What key to add in the schema to make the field mandatory?
- A. mandatory
- B. required
- C. Both A and B
- D. None

**Answer: B** 



#### 3. What combination of filters can we add in getAllTheatre?

- A. Name and pincode
- B. City and Name
- C. Name City and Pincode
- D. All are possible

**Answer: D** 

#### 4. Which option is being used to make a field immutable in mongoose?

- A. description
- B. name
- C. createdAt
- D. UpdatedAt

**Answer: C** 

#### 5. Which method is used to delete the database?

- A. delete
- B. deleteMany
- C. drop
- D. All of the above

**Answer: C** 



#### **Practice Code**

- Update the theatre.model.js to store number of halls that will be present in the specific theatre There may be multiple movies played under the theatre so, choose the data type accordingly.
- Write an API to add multiple hall to a theatre by sending threatre id and hall Ids.



**Thank You!** 

