

API Documentation – Events Module

DT Node.js Challenge – Task 2

1. Introduction

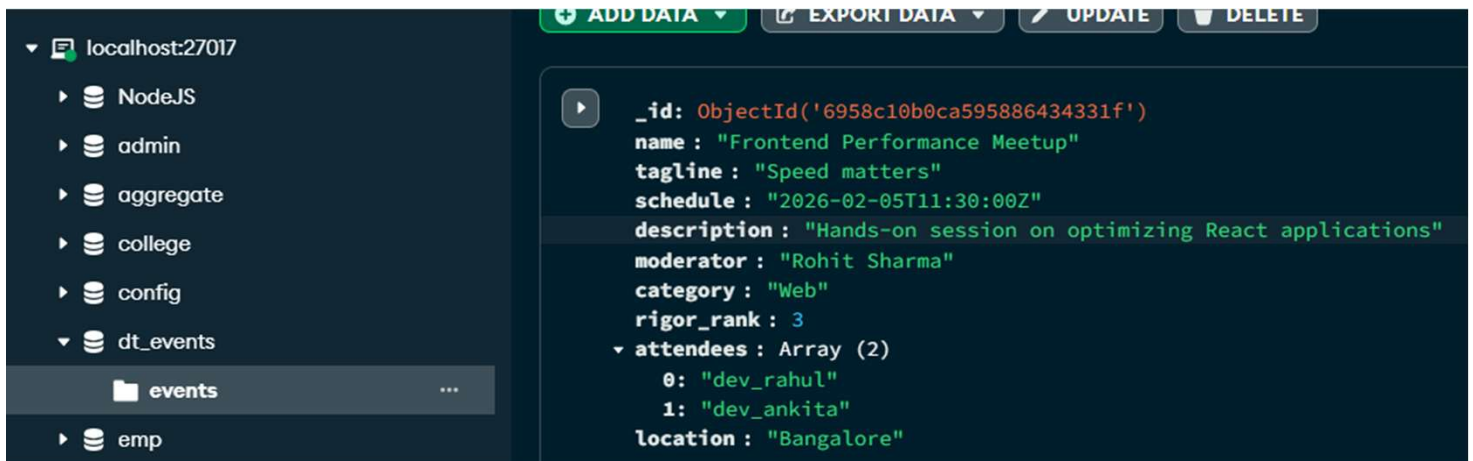
This document describes the **Events API** created in Task-1.

The API allows users to **create, read, update, and delete events** using RESTful endpoints.

The backend is implemented using **Node.js, Express.js, and MongoDB Native Driver** without enforcing schemas.

2. Base URL

<http://localhost:6000/api/v3/app/>



4. API Endpoints (CRUD Operations)

4.1 Create Event

Endpoint

POST /events

URL = <http://localhost:6000/api/v3/app/event>

Description

Creates a new event and stores it in the database.

Create Event

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

- Task:** task 1 / postEvent
- Method:** POST
- URL:** http://localhost:6000/api/v3/app/event
- Body:**

```
{
  "event": "DevOps CI/CD Pipeline Demo",
  "tagline": "Automate everything",
  "schedule": "2026-02-22T12:00:00Z",
  "category": "DevOps",
  "tools_used": ["GitHub Actions", "Docker"],
  "difficulty": "Advanced"
}
```
- Response:** 201 Created, 68 ms, 321 B. The response body is:

```
{
  "success": true,
  "message": "event inserted",
  "event_id": "695a47149f91bcd20d6f6a08"
}
```

Event is inserted in db through post event

```
{
  "event": "DevOps CI/CD Pipeline Demo",
  "tagline": "Automate everything",
  "schedule": "2026-02-22T12:00:00Z",
  "category": "DevOps",
  "tools_used": ["GitHub Actions", "Docker"],
  "difficulty": "Advanced"
}
```

4.2 Get Event by ID

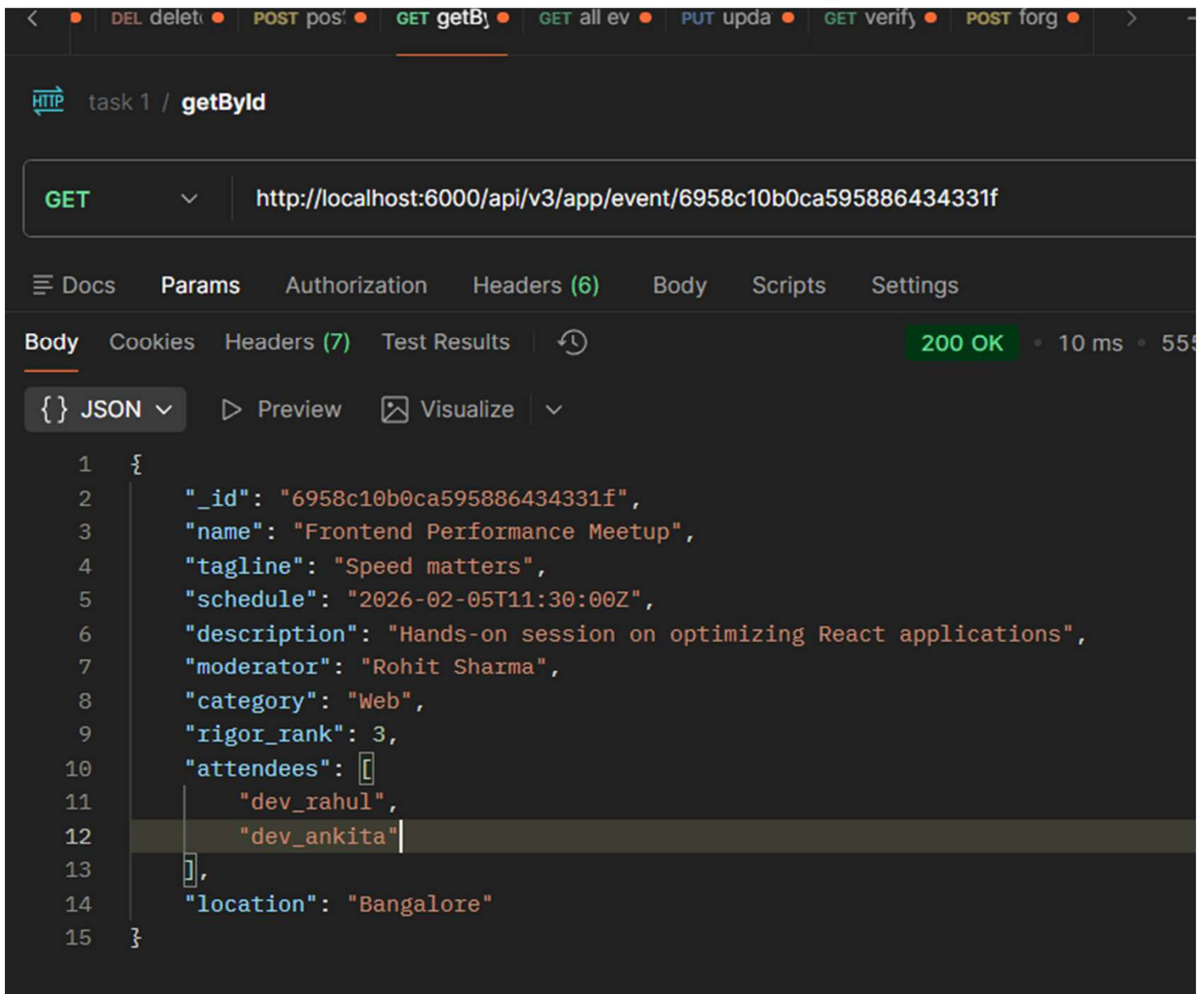
Endpoint

GET /events/:id

<http://localhost:6000/api/v3/app/event/6958c10b0ca595886434331f>

Description

Fetches a single event using its MongoDB ObjectId.



The screenshot shows a REST client interface with a tab labeled 'task 1 / getById'. The method is set to 'GET' and the URL is 'http://localhost:6000/api/v3/app/event/6958c10b0ca595886434331f'. The response is a 200 OK status with a 10 ms response time. The response body is displayed in JSON format, showing the details of an event.

```
1  {
2    "_id": "6958c10b0ca595886434331f",
3    "name": "Frontend Performance Meetup",
4    "tagline": "Speed matters",
5    "schedule": "2026-02-05T11:30:00Z",
6    "description": "Hands-on session on optimizing React applications",
7    "moderator": "Rohit Sharma",
8    "category": "Web",
9    "rigor_rank": 3,
10   "attendees": [
11     "dev_rahul",
12     "dev_ankita"
13   ],
14   "location": "Bangalore"
15 }
```

4.3 Get Latest Events

Endpoint

GET /events?type=latest&limit=5&page=1

<http://localhost:6000/api/v3/app/event/all?page=1&limit=5>

Description

Returns the most recent events with pagination support.

Query Parameters

Parameter	Description
type	latest
limit	Number of events
page	Page number

task 1 / all event

GET <http://localhost:6000/api/v3/app/event/all?page=1&limit=5>

Docs Params Authorization Headers (6) Body Scripts Settings

Query Params

<input checked="" type="checkbox"/>	Key	Value	Description
<input checked="" type="checkbox"/>	page	1	
<input checked="" type="checkbox"/>	limit	5	

Body Cookies Headers (7) Test Results 200 OK • 30 ms • 1.42 KB •

{ } JSON Preview Visualize

```
1  [
2    {
3      "_id": "6958c10b0ca595886434331f",
4      "name": "Frontend Performance Meetup",
5      "tagline": "Speed matters",
6      "schedule": "2026-02-05T11:30:00Z",
7      "description": "Hands-on session on optimizing React applications",
8      "moderator": "Rohit Sharma",
9      "category": "Web",
10     "rigor_rank": 3,
11     "attendees": [
12       "dev_rahul",
13       "dev_ankita"
14     ],
15     "location": "Bangalore"
16   },
17   {
18     "_id": "6958c1190ca5958864343320",
19     "name": "Cloud Basics Bootcamp",
```

4.4 Update Event

Endpoint

PUT /events/:id

<http://localhost:6000/api/v3/app/event/6958b85493f0ab3444d5bcd7>

Description

Updates an existing event based on the provided ID.

Payload

Same structure as Create Event (partial updates allowed).

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

- Task:** task 1 / update event
- Method:** PUT
- URL:** http://localhost:6000/api/v3/app/event/6958b85493f0ab3444d5bcd7
- Body Type:** raw (selected)
- Request Body (JSON):**

```
1 {  
2   "name": "dt task 1"  
3 }
```
- Response Status:** 201 Created
- Response Body (JSON):**

```
1 {  
2   "success": true,  
3   "message": "event updated"  
4 }
```

4.5 Delete Event


Endpoint

DELETE /events/:id

<http://localhost:6000/api/v3/app/event/6958b85493f0ab3444d5bcd7>

Description

Deletes an event using its ObjectId.

 task 1 / **delete event**

DELETE

▼

http://localhost:6000/api/v3/app/event/6958b85493f0ab3444d5bcd7

☰ Docs

Params

Authorization

Headers (6)

Body

Scripts

Settings

Query Params


	Key	Value	Description
	Key	Value	Description

Body

Cookies

Headers (7)


Test Results



200 OK • 76 ms •

{ } JSON ▼

▶ Preview

 Visualize ▼

1

2

3

4

[

"success": true,

"message": "event deleted"

]

5. Directions and Limitations

MongoDB native driver is used

No Mongoose or predefined schema

Flexible and dynamic data structure

_id is used as the unique identifier

Supports querying on any field

6. Why No Schema

The API is designed to be **schema-less** to allow flexibility.

This approach helps in adding or removing fields without changing database structure and avoids tight coupling to a fixed schema.