

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/>

The screenshot shows a MongoDB interface with a sidebar containing database and collection names: NodeJS, admin, aggregate, college, config, dt_events, events (selected), and emp. The main area displays a document with the following fields and values:

```
_id: ObjectId('6958c10b0ca595886434331f')
name : "Frontend Performance Meetup"
tagline : "Speed matters"
schedule : "2026-02-05T11:30:00Z"
description : "Hands-on session on optimizing React applications"
moderator : "Rohit Sharma"
category : "Web"
rigor_rank : 3
attendees : Array (2)
  0: "dev_rahul"
  1: "dev_ankita"
location : "Bangalore"
```

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 the Postman application interface. At the top, it says "task 1 / postEvent". Below that, a "POST" method is selected for the request type, and the URL is set to "http://localhost:6000/api/v3/app/event". The "Body" tab is active, showing a JSON payload:

```
1 {  
2   "event": "DevOps CI/CD Pipeline Demo",  
3   "tagline": "Automate everything",  
4   "schedule": "2026-02-22T12:00:00Z",  
5   "category": "DevOps",  
6   "tools_used": ["GitHub Actions", "Docker"],  
7   "difficulty": "Advanced"  
8 }  
9
```

Below the body, the response details are shown: "201 Created" status, "68 ms" duration, and "321 B" size. The response body is also displayed in JSON format:

```
{ } JSON ▾ ▶ Preview ▶ Visualize ▾
```

```
1 {  
2   "success": true,  
3   "message": "event inserted",  
4   "event_id": "695a47149f91bcd20d6f6a08"  
5 }
```

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

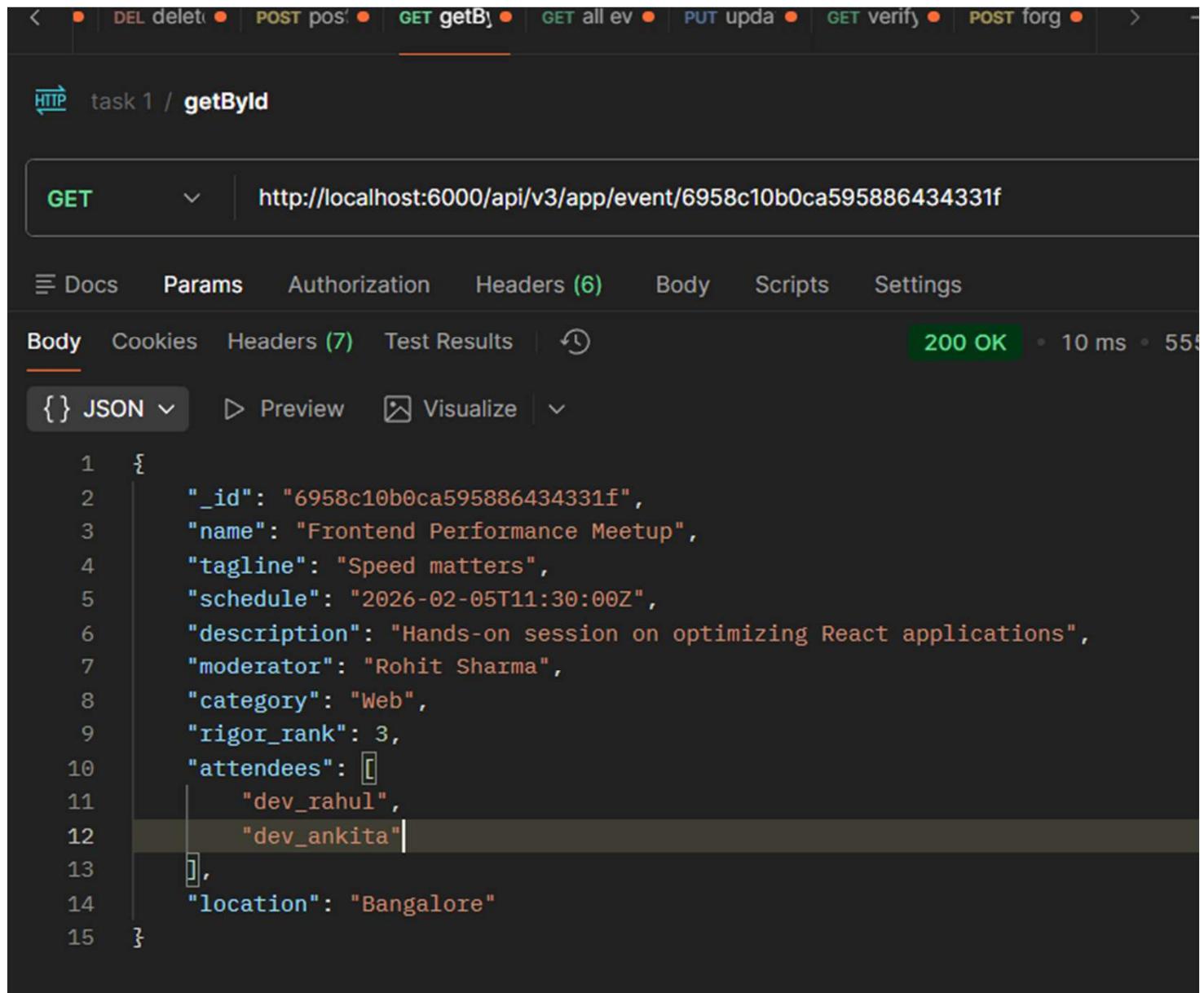
Endpoint

GET /events/:id

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

Description

Fetches a single event using its MongoDB ObjectId.



The screenshot shows a Postman collection named "task 1" with a single endpoint named "getById". The request method is GET, and the URL is <http://localhost:6000/api/v3/app/event/6958c10b0ca595886434331f>. The response status is 200 OK, with a response time of 10 ms and a size of 558 bytes. The response body is a JSON object representing the 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

The screenshot shows the Postman application interface. At the top, it displays the URL `http://localhost:6000/api/v3/app/event/all?page=1&limit=5`. Below the URL, there are tabs for 'Docs', 'Params' (which is selected), 'Authorization', 'Headers (6)', 'Body', 'Scripts', and 'Settings'. Under the 'Params' tab, there is a table with three rows: 'page' (value 1) and 'limit' (value 5). In the 'Body' tab, the response is shown as JSON. The JSON data consists of two objects, each representing an event. The first event has an _id of "6958c10b0ca595886434331f", a name of "Frontend Performance Meetup", a tagline of "Speed matters", a schedule of "2026-02-05T11:30:00Z", a description of "Hands-on session on optimizing React applications", a moderator of "Rohit Sharma", a category of "Web", a rigor_rank of 3, attendees including "dev_rahul" and "dev_ankita", and a location of "Bangalore". The second event has an _id of "6958c1190ca5958864343320" and a name of "Cloud Basics Bootcamp". The status bar at the bottom right indicates a 200 OK response with a time of 30 ms and a size of 1.42 KB.

Key	Value	Description
page	1	
limit	5	

```
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",  
20 }]
```

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 the Postman application interface. At the top, it says "task 1 / update event". Below that, the method is set to "PUT" and the URL is "http://localhost:6000/api/v3/app/event/6958b85493f0ab3444d5bcd7". The "Body" tab is selected, showing the raw JSON payload:

```
1 {  
2   "name": "dt task 1"  
3 }
```

Below the body, the "Body" tab is again selected, along with "Cookies", "Headers (7)", and "Test Results". The status bar indicates "201 Created" and "21". The "Body" section shows the response JSON:

```
{ } 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.

The screenshot shows a Postman task named "task 1 / delete event". The request method is set to "DELETE" and the URL is "http://localhost:6000/api/v3/app/event/6958b85493f0ab3444d5bcd7". The "Params" tab is selected, showing a table with one row and two columns: "Key" and "Value". The "Body" tab is selected, showing a JSON response with the following content:

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

The response status is "200 OK" and the execution time is "76 ms".

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.