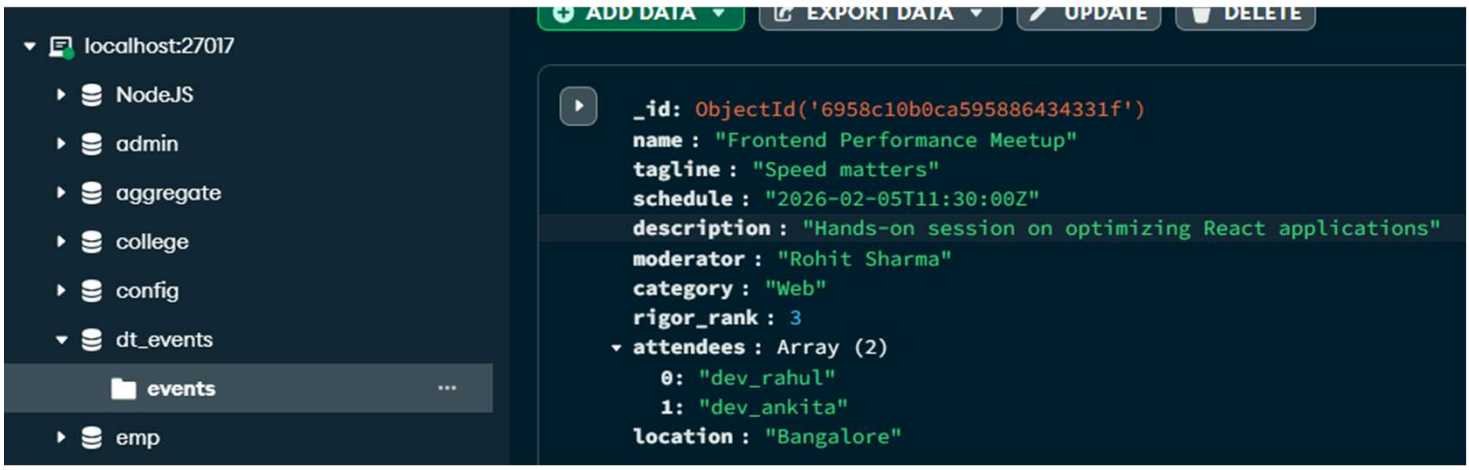
API Documentation – Events Module
DT Node.js Challenge – Task 2

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/

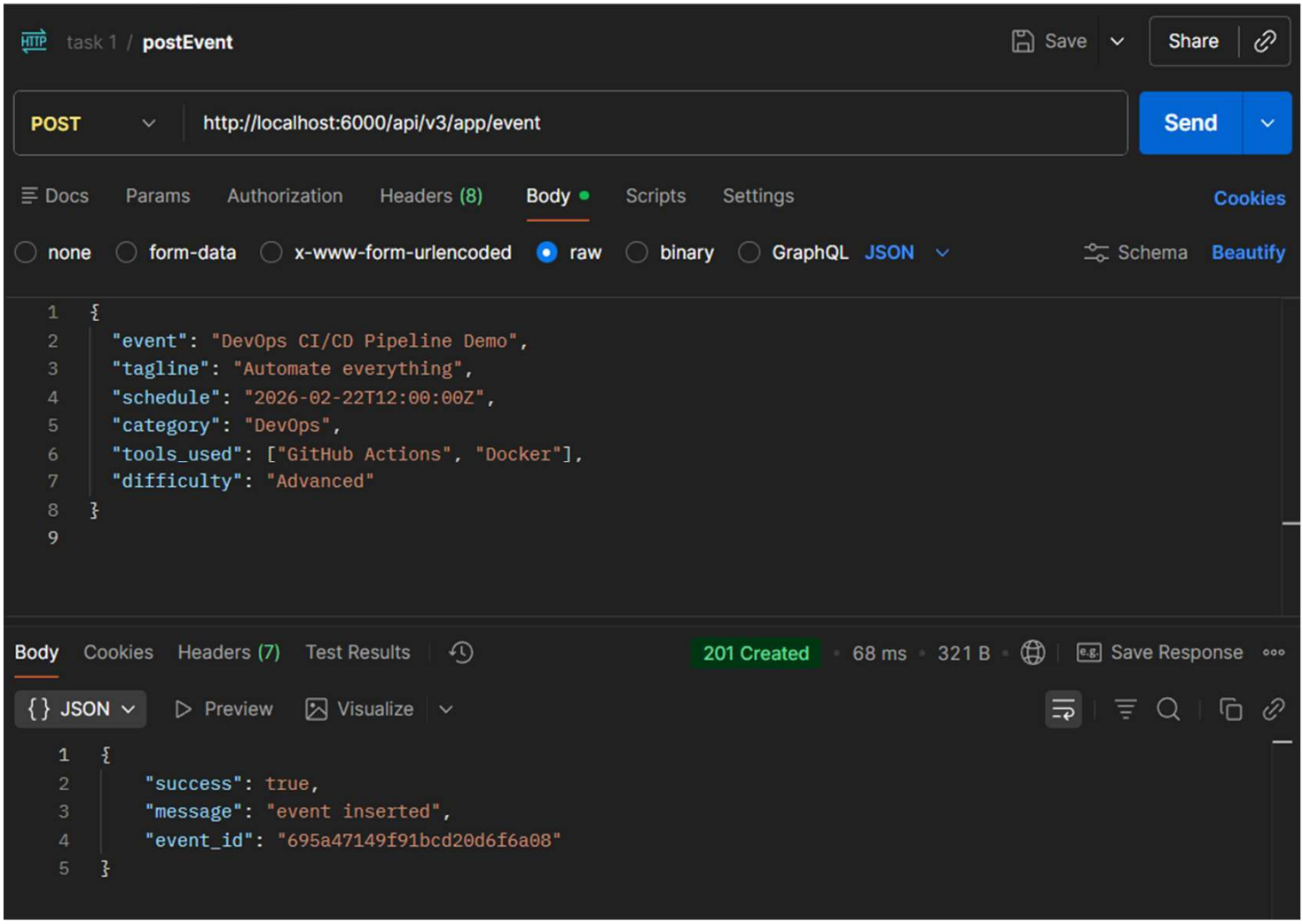


4. API Endpoints (CRUD Operations)

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



Event is inseterted 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

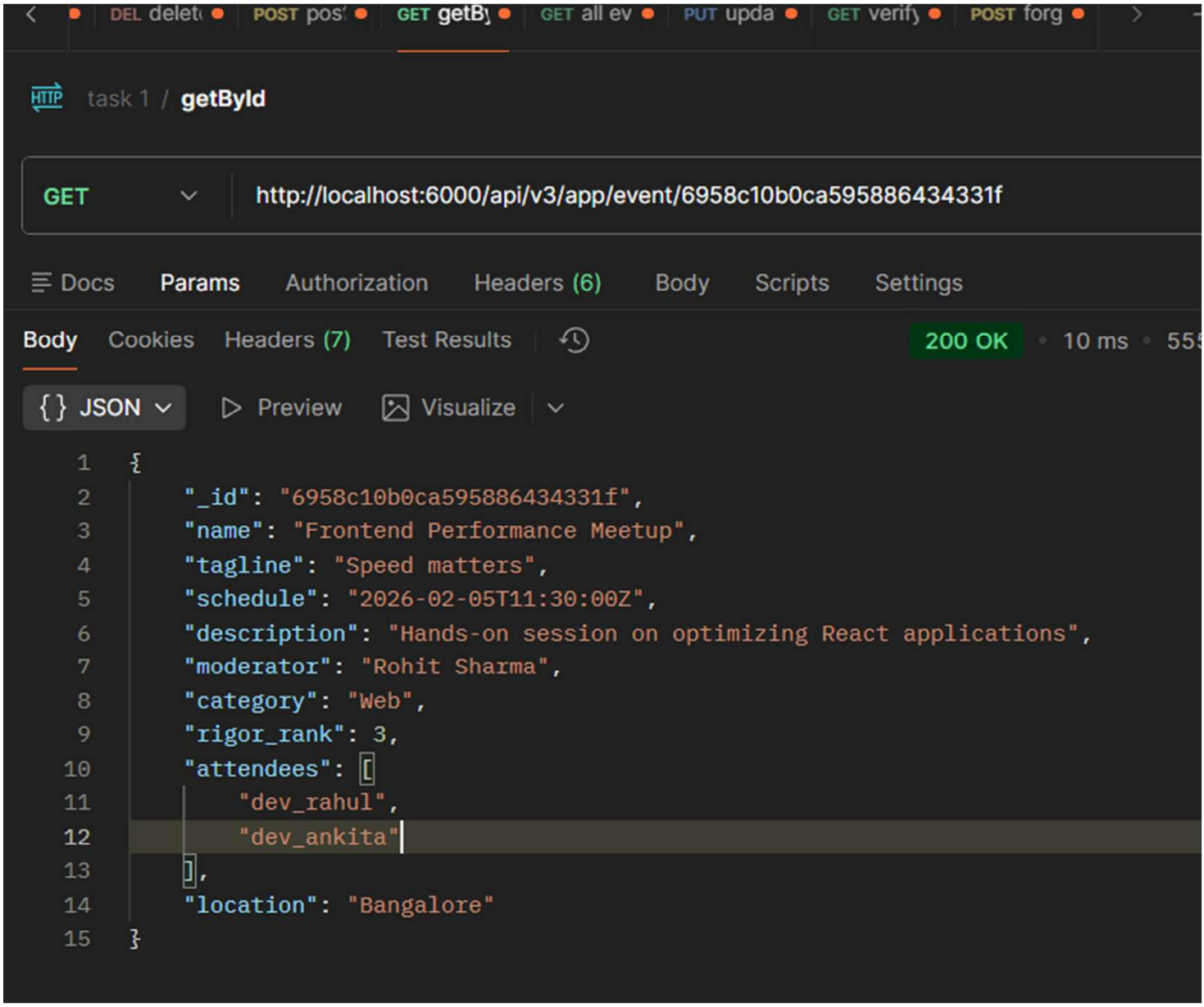
Endpoint

GET /events/:id

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

Description

Fetches a single event using its MongoDB ObjectId.



4.3 Get Latest Events

Endpoint

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

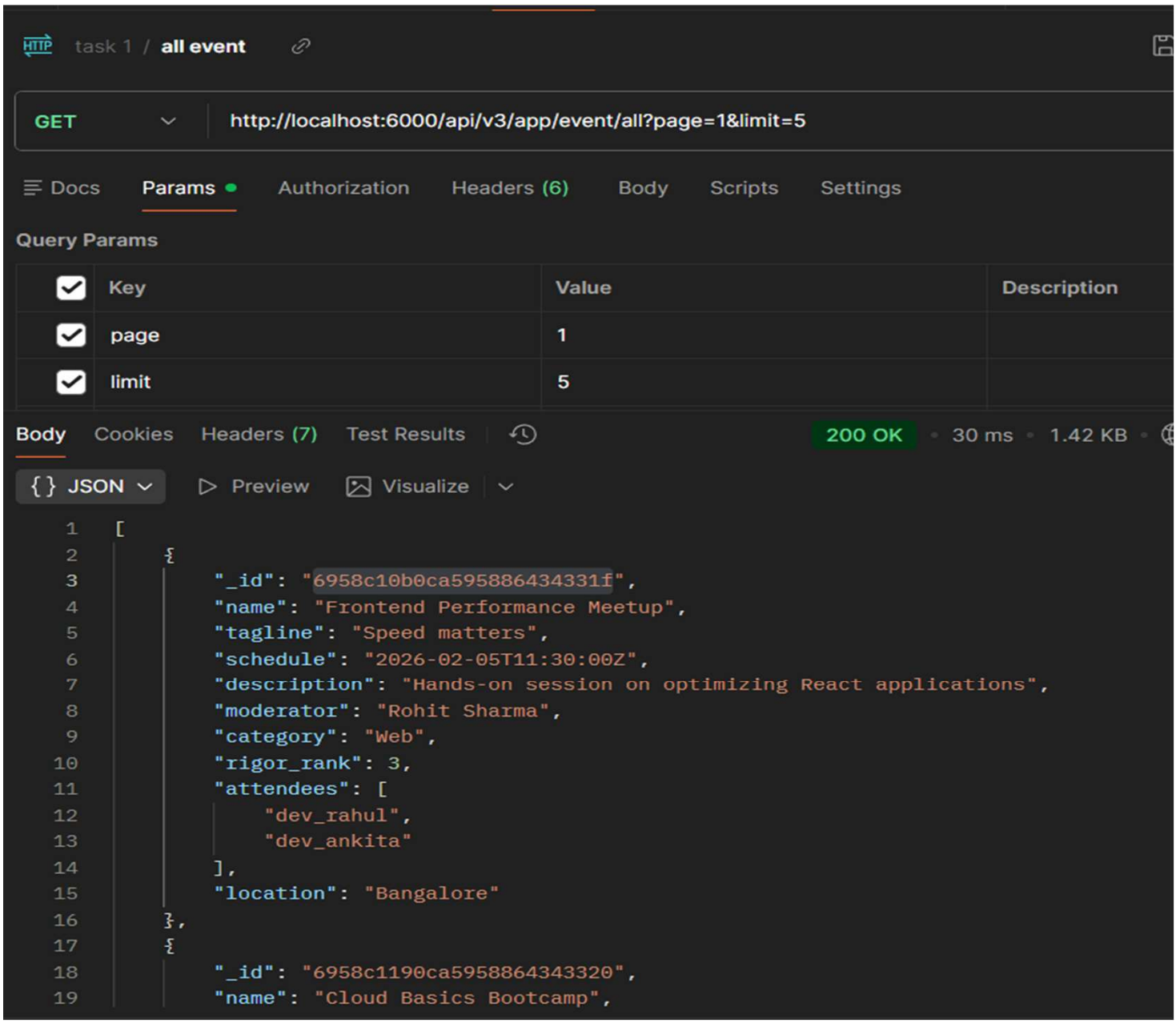
http://localhost:6000/api/v3/app/event/all?page=1&limit=5

Description

Returns the most recent events with pagination support.

Query Parameters

Parameter	Description
type	latest
limit	Number of events
page	Page number



4.4 Update Event

Endpoint

PUT /events/:id

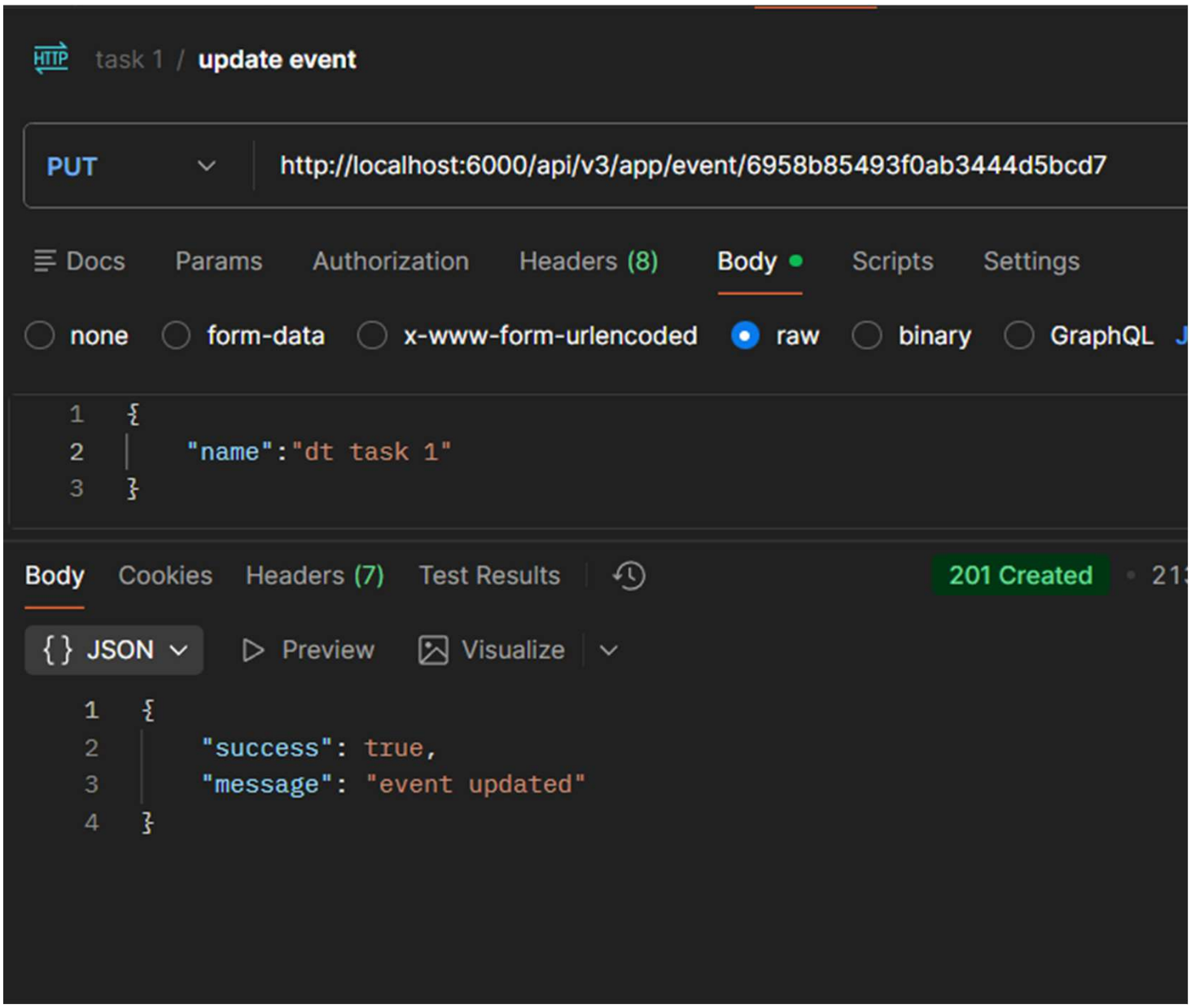
http://localhost:6000/api/v3/app/event/6958b85493f0ab3444d5bcd7

Description

Updates an existing event based on the provided ID.

Payload

Same structure as Create Event (partial updates allowed).



4.5 Delete Event

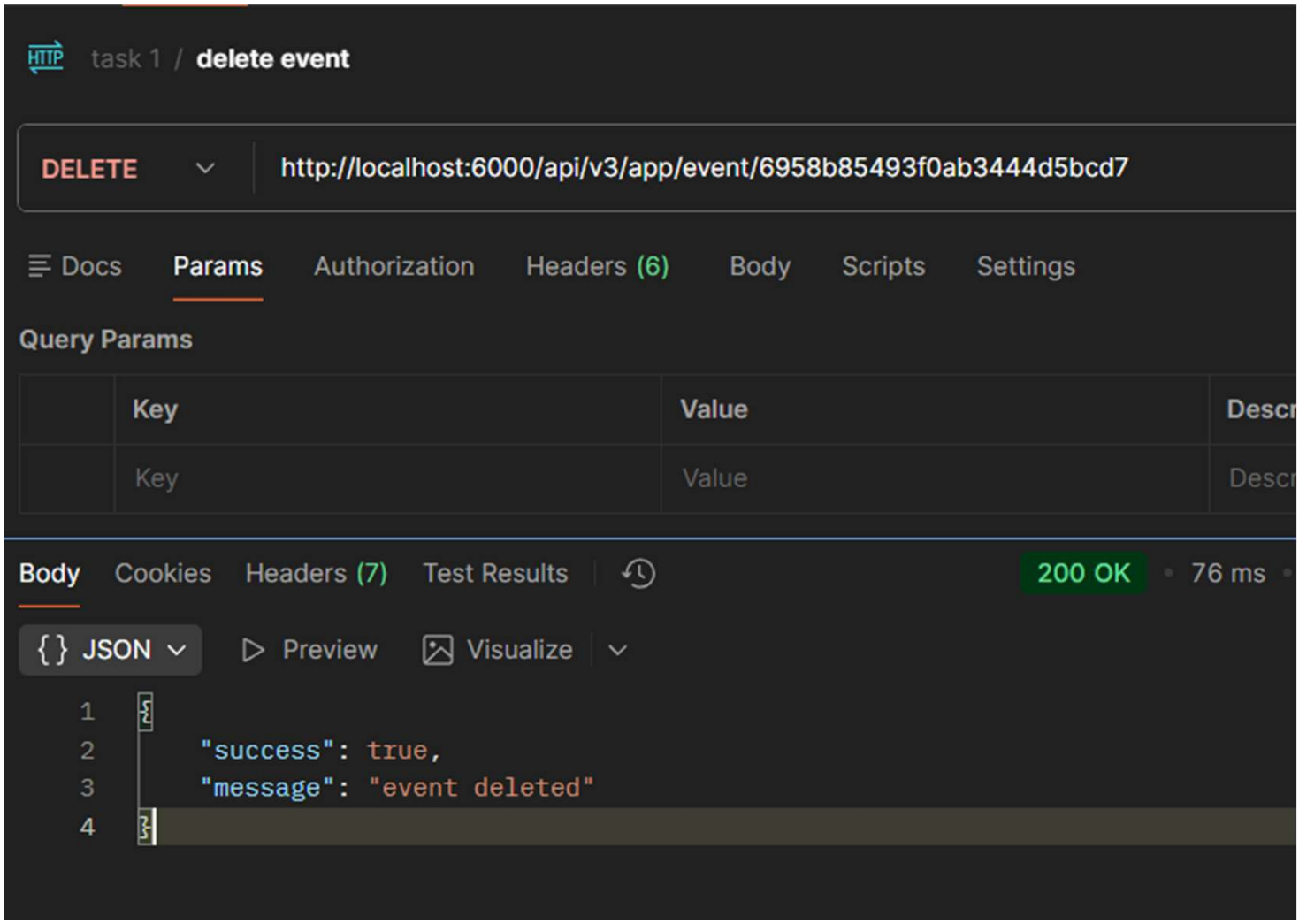
Endpoint

DELETE /events/:id

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

Description

Deletes an event using its ObjectId.



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.