## **Project Report: Task Management API**

**Objective**

The goal of this project is to build a secure, RESTful API using Node.js and Express that allows users to manage a list of tasks. The system supports task creation, retrieval, updating, deletion, and searching, all behind JWT-based authentication.

**Architecture & Design Approach**

1. **Backend Stack**:
   * **Node.js** for runtime
   * **Express.js** for routing and middleware
   * **UUID** for generating unique task IDs
   * **Morgan** for request logging
   * **JSON Web Tokens (JWT)** for authentication and route protection
2. **Data Handling**:
   * Data is stored in-memory as an array of task objects, each with a unique id, title, and description.
3. **RESTful Principles**:
   * Each route corresponds to a standard HTTP verb (GET, POST, PUT, DELETE).
   * Resources (/tasks) are accessed and manipulated via predictable URLs.

**Security & Authentication**

* A /login route simulates user login by issuing a JWT token.
* All /tasks routes are protected by a middleware (authenticateToken) which validates the JWT.
* Tokens are passed via the Authorization header in the format Bearer <token>.

**Middleware Strategy**

* authenticateToken: Ensures the request is coming from an authenticated user.
* validateTask: Confirms that both title and description fields are present in requests modifying tasks.

**Core Features**

1. **Create Task**:
   * Validates input, generates a UUID, adds the task to memory.
2. **Get All Tasks**:
   * Supports pagination (page, limit), filtering (search), and sorting (sortBy, order).
   * Implements case-insensitive search and flexible field-based sorting.
3. **Get Task by ID**:
   * Finds a task matching the id in the URL; returns 404 if not found.
4. **Update Task**:
   * Checks for existence, then merges existing task with new data.
5. **Delete Task**:
   * Locates the task by ID and removes it from the array.

**How to Run:**

1. Install dependencies:

npm install

2. Start the server:

node app.js

3. Use Postman or curl to test the API.

- First, POST to /login with a JSON body:

{

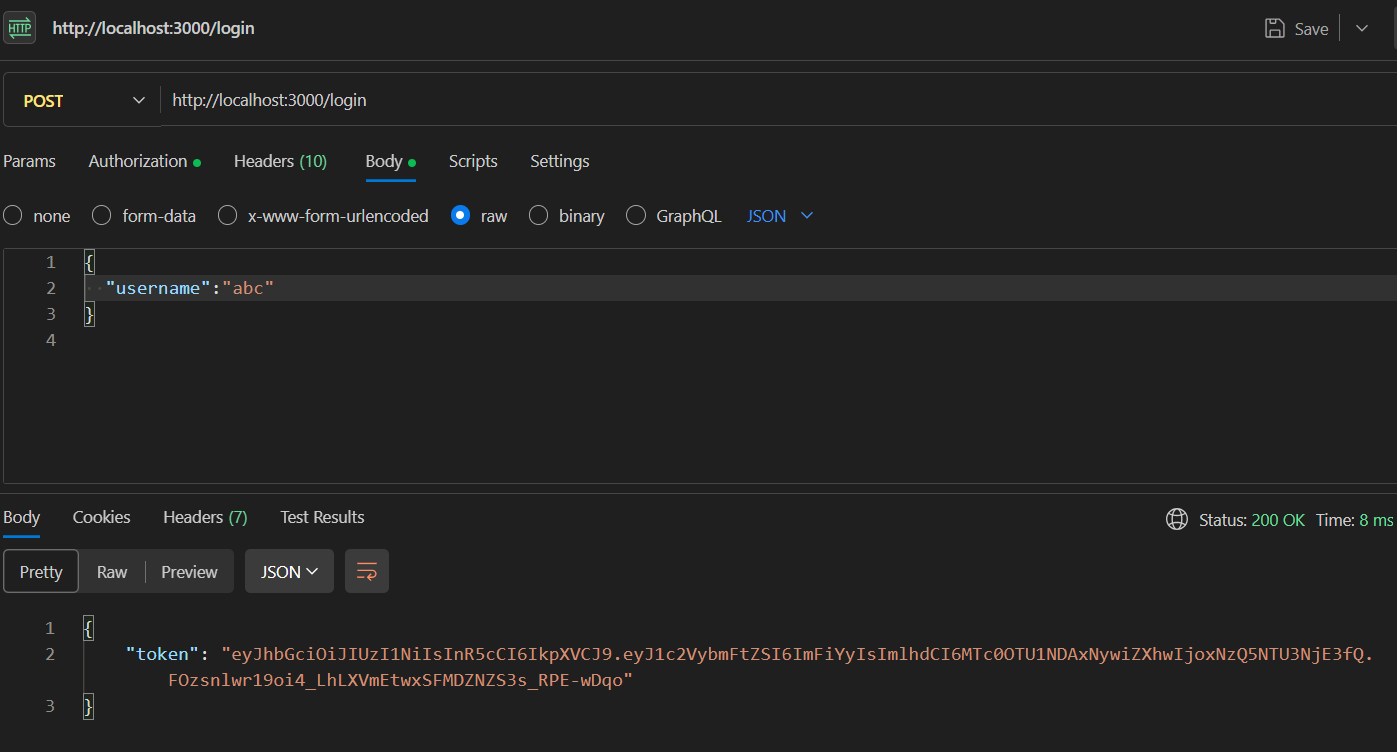
"username": "your-name"

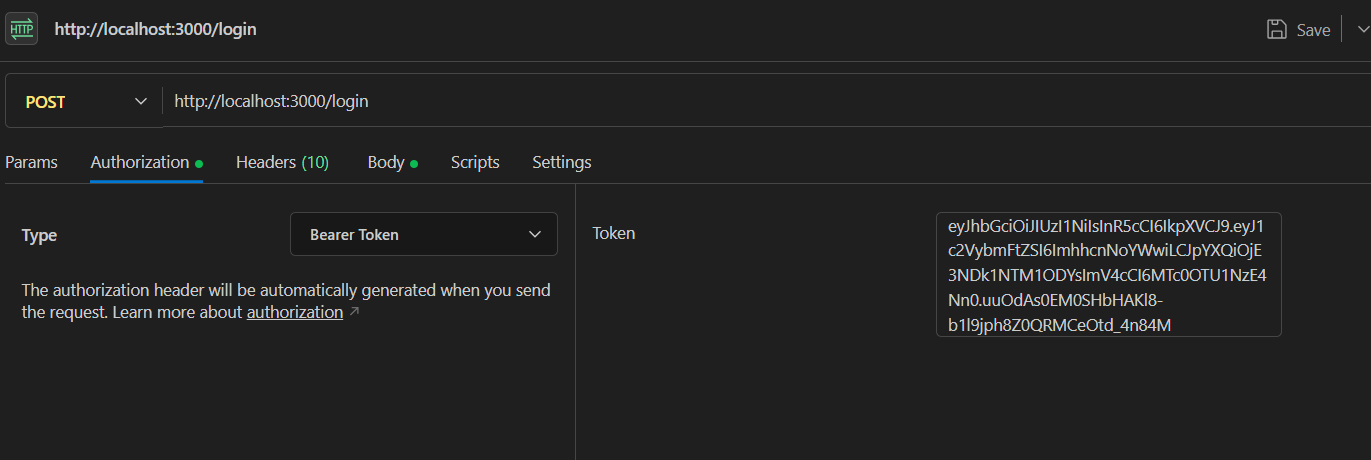
}

- Use the returned token in the Authorization header for all /tasks requests.

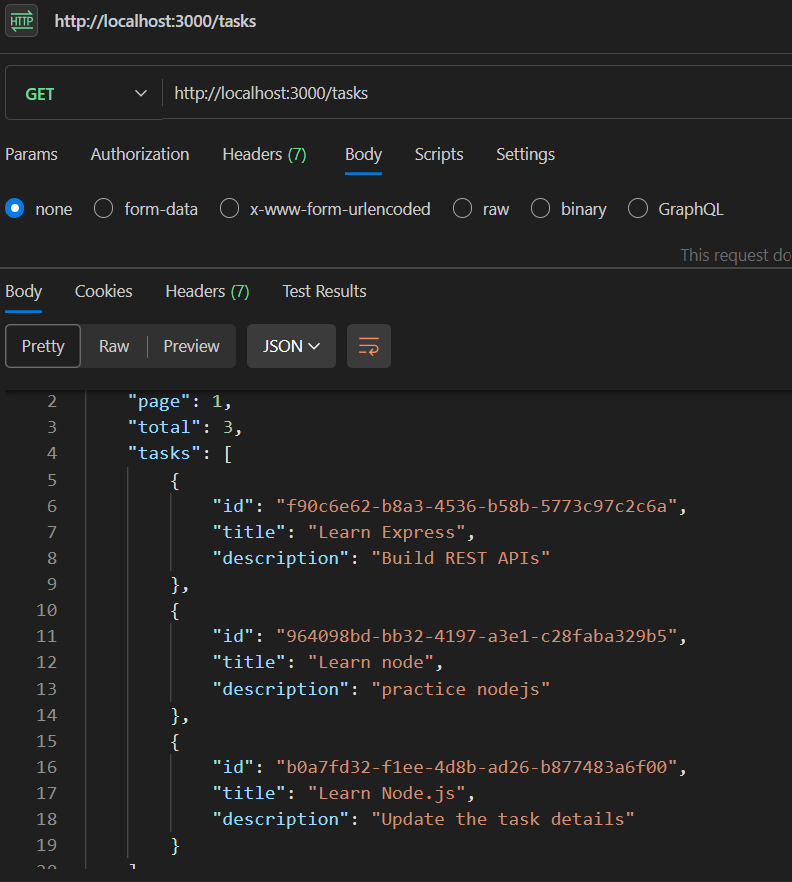
**API Endpoints:**

1. JWT Authentication- login

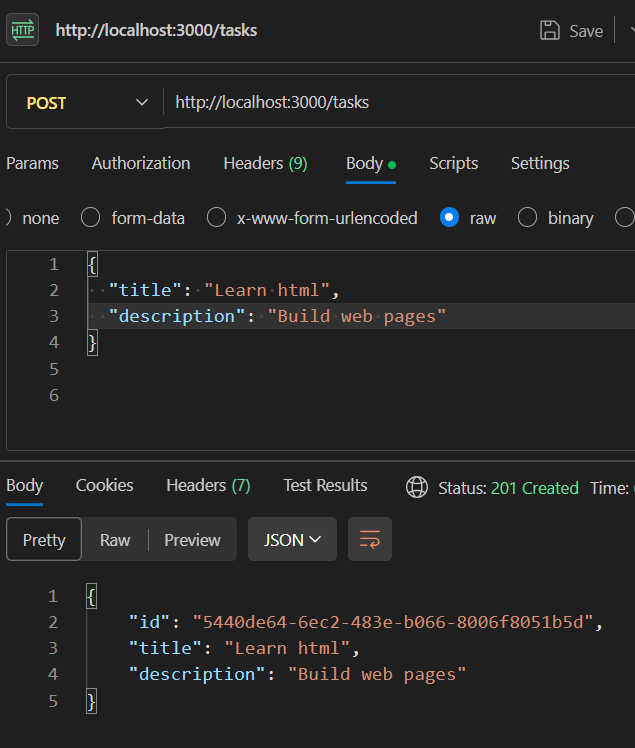




1. Get - http://localhost:3000/tasks

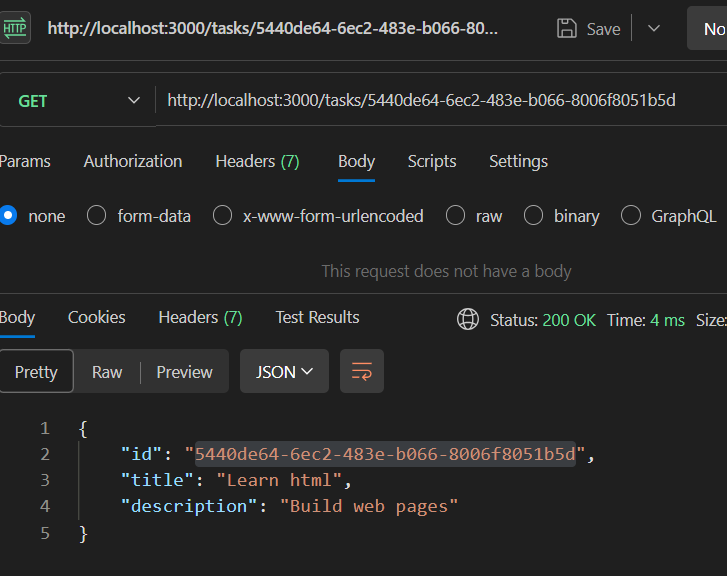


1. Post - http://localhost:3000/tasks



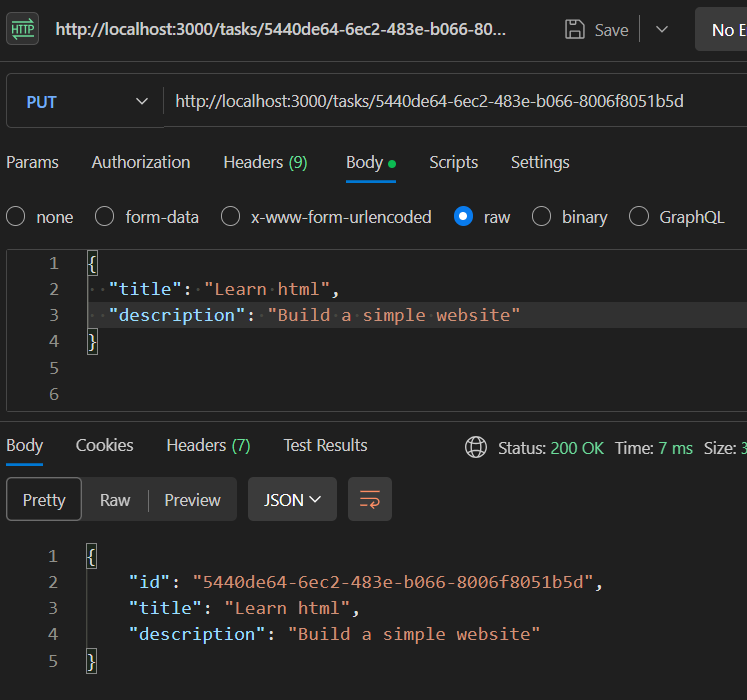
1. Get by id –

http://localhost:3000/tasks/5440de64-6ec2-483e-b066-8006f8051b5d



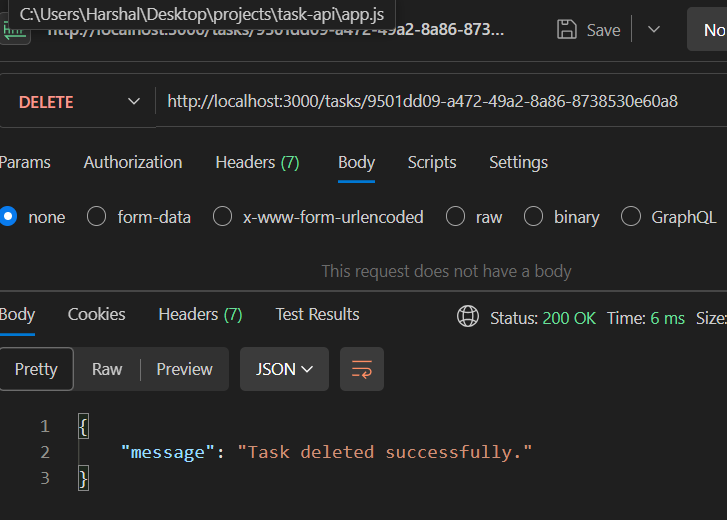
1. PUT /tasks/:id

http://localhost:3000/tasks/5440de64-6ec2-483e-b066-8006f8051b5d



1. DELETE /tasks/:id

http://localhost:3000/tasks/9501dd09-a472-49a2-8a86-8738530e60a8



1. Get All Tasks (with pagination, filtering)