# **Node.js Developer Test Task**

Welcome to the KeyM hiring process for Node.js developers! This test task is designed to assess your backend development skills using Node.js with either Express.js or NestJS.

## **Task Description**

You will create a simple backend service for managing a booking system. The service should expose a REST API (or GraphQL if preferred) that allows users to book time slots. Follow the requirements and guidelines below.

### **Requirements**

#### **1. API Endpoints**

Your service should support the following functionality:

* **Create a Booking**
  + Endpoint: POST /bookings

Request Body:  
{

"user": "User's name",

"date": "YYYY-MM-DD",

"startTime": "HH:mm",

"endTime": "HH:mm"

}

* + Response: Return the created booking with a unique ID.
  + **Important:** Ensure that there is no overlap of slots for the same date and time range.
* **Get All Bookings**
  + Endpoint: GET /bookings
  + Response: Return a list of all bookings.
* **Get a Booking by ID**
  + Endpoint: GET /bookings/:id
  + Response: Return the booking with the specified ID.
* **Delete a Booking**
  + Endpoint: DELETE /bookings/:id
  + Response: Confirm deletion.

#### **2. Data Validation**

* Validate incoming request data using a library such as Joi or built-in validation from NestJS.
* Ensure all required fields are present and in the correct format.
* Validate that startTime is earlier than endTime and the time slot is within the same day.

#### **3. Conflict Handling**

* Implement logic to ensure no overlapping bookings for the same date and time range.
* Return an appropriate error message if a conflict is detected.

#### **4. Storage**

* Use any of the following options for data storage:
  + In-memory storage (e.g., an array or Map).
  + MongoDB (preferred), with Mongoose or Prisma.
  + SQLite or PostgreSQL.

#### **5. Documentation**

* Provide API documentation using Swagger (or another tool) to describe all endpoints and their request/response formats.

#### **6. Code Quality**

* Ensure the code is clean, well-structured, and follows best practices.
* Include meaningful comments and error handling.

#### **7. Testing**

* Write unit tests for your API endpoints using Jest, Mocha, or another testing framework.
* Cover at least 70% of the codebase.

#### **8. Bonus Points**

* Add a feature to update an existing booking.
* Deploy the application to a free cloud service (e.g., Render, Vercel, or Railway) and provide the live URL.
* Include authentication using JWT.

### **Submission Guidelines**

1. Push your code to a public GitHub repository.
2. Include a README.md file with:
   * Instructions on how to run the project locally.
   * A list of features implemented.
   * Any additional notes or assumptions made.
3. Submit the link to your GitHub repository and your CV using the [submission form](https://calendly.com/career-key-m/nodejs-interview).

### **Evaluation Criteria**

* Functionality: Does the service meet the requirements?
* Code Quality: Is the code clean, modular, and maintainable?
* Testing: Are the tests meaningful, and do they provide good coverage?
* Documentation: Is the API well-documented and easy to understand?
* Bonus Features: Have any additional features been implemented?