

# Web Development Internship - Task 3

## REST API for Book Management

Soumen Das

November 17, 2025

### 1. Objective

The objective of this task is to create a simple REST API to manage a list of books using Node.js and Express. The API provides endpoints for CRUD (Create, Read, Update, Delete) operations. The data is stored in memory in a simple array, and no database is required.

### 2. How to Run the Server

To run this project, you need to have Node.js and npm installed.

1. **Save the Code:** Save the provided server code as `index.js` in a new folder.
2. **Initialize Project:** Open a terminal in that folder and run `npm init -y` to create a `package.json` file.
3. **Install Dependencies:** Install Express and body-parser by running:  
`npm install express body-parser`
4. **Start the Server:** Run the server with the following command:  
`node index.js`
5. **Test:** The server will be running on `http://localhost:3000`. You can now test the endpoints using a tool like Postman.

### 3. API Endpoints

The following endpoints are available on the server:

### 4. Outcome

This task successfully demonstrates the understanding of REST API basics, Express routing, different HTTP methods (GET, POST, PUT, DELETE), and JSON handling for request and response bodies.

| Method | Endpoint   | Description  |
|--------|------------|--|
| GET    | /books     | Retrieves the full list of all books.  |
| GET    | /books/:id | Retrieves a single book specified by its <code>id</code> .   |
| POST   | /books     | Adds a new book to the list.<br><i>Requires a JSON body: {"title": "...", "author": "..."} </i>                                |
| PUT    | /books/:id | Updates an existing book specified by its <code>id</code> .<br><i>Requires a JSON body: {"title": "...", "author": "..."} </i> |
| DELETE | /books/:id | Deletes a book specified by its <code>id</code> .  |

Table 1: Available REST API Endpoints