Description.md

To-Do List Management API Documentation

Overview

This API provides endpoints for managing a to-do list, including functionalities for adding, editing, deleting tasks, marking tasks as completed, and retrieving tasks. It is built using NestJS and MongoDB with Mongoose for data storage.

Code Structure

Source Directory (`src`)

- **`app.controller.ts`**: Controller for handling incoming requests to the root endpoint.
- **`app.controller.spec.ts`**: Unit tests for `app.controller.ts`.
- **`app.module.ts`**: Main module for the application, imports other modules.
- **`app.service.ts`**: Provides services to the controller.
- **`main.ts`**: Entry point of the application.
- **`tasks/`**: Directory containing task-related modules, controllers, and services.
  - **`task.model.ts`**: Mongoose schema and interface for the Task model.
  - **`tasks.controller.ts`**: Controller for handling task-related requests.
  - **`tasks.service.ts`**: Provides task-related business logic.

Test Directory (`test`)

- **`app.controller.e2e-spec.ts`**: End-to-end tests for the `AppController`.
- **`tasks.controller.e2e-spec.ts`**: End-to-end tests for the `TasksController`.
- **`jest-e2e.json`**: Configuration for end-to-end testing with Jest.

Key Decisions

1. **NestJS Framework**: Chosen for its modular architecture, built-in dependency injection, and powerful CLI.
2. **Mongoose**: Used for MongoDB object modeling, providing schema-based solutions for application data.
3. **Modular Structure**: Organized code into modules (e.g., `tasks` module) to enhance maintainability and scalability.
4. **Testing**: Included both unit and end-to-end tests to ensure code quality and reliability.

5. **Environment Variables**: Used environment variables to store sensitive information such as MongoDB connection strings.

API Endpoints

`Task Endpoints`

Get All Tasks

- **URL**: `/tasks`
- **Method**: GET
- **Description**: Retrieves all tasks from the database.
- **Response**: Array of Task objects.

Create a Task

- **URL**: `/tasks`
- **Method**: POST
- **Description**: Creates a new task.
- **Request Body**:

```
{
  "title": "Task Title",
  "planning": "Task Planning",
  "done": false
}
```

- **Response**: The created Task object.

Update a Task

- **URL**: `/tasks/:id`
- **Method**: PUT
- **Description**: Updates an existing task by ID.
- **Request Body**:

```
{
  "title": "Updated Task Title",
  "planning": "Updated Task Planning",
  "done": false
}
```

- **Response**: The updated Task object.

Delete a Task

- **URL**: `/tasks/:id`
- **Method**: DELETE
- **Description**: Deletes a task by ID.
- **Response**: The deleted Task object.

Mark a Task as Done

- **URL**: `/tasks/:id/done`
- **Method**: PATCH
- **Description**: Marks a task as completed by ID.
- **Response**: The updated Task object with `done` set to `true`.

Running the Application

Prerequisites

- Node.js and npm
- MongoDB instance

Installation

1. Clone the repository:

   ```
   git clone https://github.com/itsarraj/todo-api-nestjs.git
   ```

2. Install dependencies:

   ```
   cd todo-api-nestjs
   npm install
   ```

Running the Application

1. Start the NestJS application:

   ```
   npm run start
   ```

2. The application will be running at `http://localhost:3000`.

Running Tests

- Unit tests:

  ```
  npm run test
  ```

- End-to-end tests:

```
npm run test:e2e
```

This documentation provides an overview of the code structure, key decisions, and API endpoints for the To-Do List Management API built with NestJS. For further details, refer to the code comments and the respective modules.