Senior MERN stack Interview task

MERN task

Objective

To evaluate the candidate’s ability to:

- Build a well-structured **Node.js backend with Express**.
- Write **efficient and scalable APIs**.
- Implement **authentication and authorization**.
- Design **logical solutions** for real-world problems.
- Use **MongoDB effectively** with proper indexing and queries.
- Handle **edge cases and performance optimizations**.

Task: Multi-Level Category Management API

Problem Statement

You need to build a **category management API** where:

- Categories can be **nested** (like a tree structure).
- Each category has a **name, parent category (optional), and status (active/inactive)**.
- Categories should be **retrievable in a tree structure**.
- API should support **CRUD operations**.
- When a **parent category is deleted, all child categories should be reassigned** to its parent.
- When a category is **marked inactive, all its subcategories should be inactive**.

Requirements

1. **Tech Stack**:
   - **Node.js**
   - **Express.js**
   - **Typescript only**
   - **MongoDB (Mongoose ORM)**
   - **JWT for authentication**
   - **Postman API Collection for testing (optional but preferred)**
2. **Features to Implement**
   - **User Authentication**
     - Implement a basic **JWT-based auth system**.
     - User should be able to **register & login**.
     - Protect API routes using authentication.
   - **Category API**
     - **Create a category** (with optional parent).
     - **Fetch all categories in a tree format**.
     - **Update a category** (name or status).
     - **Delete a category** (reassign subcategories to its parent).
   - **Performance Considerations**
     - Use proper **indexes** on MongoDB.
     - Optimize **queries** to avoid unnecessary deep lookups.
     - Handle **bulk updates efficiently** when changing status.
3. **API Routes Sample**

```
POST /api/auth/register  → Register a user
POST /api/auth/login  → Login user and return JWT token
POST /api/category  → Create a new category (Auth required)
GET /api/category  → Fetch all categories as a tree (Auth required)
PUT /api/category/:id  → Update category (Auth required)
DELETE /api/category/:id  → Delete category & reassign subcategories (Auth required)
```

Jest Test Cases (Mandatory)

- Write **unit tests** for `authController.ts` and `categoryController.ts`.
- Write **integration tests** for APIs using **Supertest**.
- Mock database interactions using **MongoMemoryServer**.
- Test authentication flows (JWT verification).
- Ensure **error handling** is tested (e.g., invalid input, unauthorized access).

Submission Guidelines

- **Upload code on GitHub** with a README.md explaining:
  - How to set up & run the project.
  - Sample API responses.
- **Bonus**: Dockerize the project.

Expected Time to Complete

3-5 hours (assuming strong MERN experience).

Good luck 🙂