Objective:
Build a RESTful API service using NestJS, capable of performing CRUD (Create, Read, Update, Delete) operations on resources of your choice (e.g, products, users) with the ability to switch between two different databases at runtime.

The resource needs to have relationships to other resources (e.g. one to one between user and user details, many to many user and roles and demonstrate a one to many relationship)

The database should be designed in such a way that the master database contains connection to clients database. The first connection the application makes is to the master database and then based on the request, the correct database is picked for a particular user.

The service should be containerized using Docker.

Requirements:
The application must be built using the NestJS framework to structure the REST API.

REST API: Implement the following endpoints with appropriate HTTP methods:

- List all resources.
- Create a new resource.
- Get a resource by ID.
- Update a resource by ID.
- Delete a resource by ID.

Database Switching: The application should support multiple different database connections. Implement functionality to switch between these databases without stopping the service. Provide configuration instructions.

Validation & Error Handling: Implement input validation for the API endpoints and proper error handling.

Dockerization: Containerize the application using Docker. Provide a Dockerfile and a docker-compose.yml (if necessary) to build and run the application.

Documentation: Include a README file with Instructions on how to build and run the application and a brief explanation of the API endpoints.

Bonus:
- Add authentication using JWT token and make use of guards
- Use Redis to cache the response when getting a resource
- Use interceptors for error logs