July 22 Backend NodeJS Exercise 1

HR Service Project Specs

Intro: This project integrates frontend via API-Gateway(nodejs) with RMQ(RabbitMQ) on PUB-SUB Model having a background HR Service.

Part 1: React APP containing Three Screens

--> Employees Listings

--> Employees Add/Edit

--> Department Listings

--> Department Add/Edit

Part 2: NodeJs API-Gateway With RMQ and Sockets

--> This application will work as gateway serving requests from Frontend and providing them service on demand basis.

--> On UI Initialization this API should hold the Socket that the Browser got connected to.

--> API should accept the request on HTTP (GET/PUT/POST/DELETE) using RequestModel, RequestModelQuery & ResponseModel

--> Its mandatory to send the socketid with requestmodel specially for PUT/POST/DELETE

--> Only for PUT/POST/DELETE the same requestModel<Specific Dto> is relayed to the RMQ on specific Topic.

** On initialization of API there needs a singleton class that needs to check all the availablity of all the RMQ Topics/Exchanges, Queues & Binding, if not there in RMQ should create the necessary Exchange/Queues/Bindings as per JSON.

--> For Get its a simple relay to the HR Service.

Part 3: HR Service

--> This service is build on NestJS using TypeORM

--> Employee, Department has one-many relationship with EmployeeDepartments.

--> There should be code written in proper OOPS and SOLID Design Principles.

--> Through Employees we should be able to insert data to EmployeeDepartments which is one-Many with Employees as well as Departments.

--> There should be proper mapping done, so that we do only expose properties via the DTO and not the Entity.

--> Here also we should have Nest Lifecycle hooks through which we will listen to specific Queues as per JSON.

--> Make sure the singleton class for Message Broker is used with callbacks so that we can use it even to push data to RMQ.

DELIVERABLES

1. UI Screen with Menu and login via google (open-id) using Authorization Code with PKCE. (on react js)
2. Validation of Token at API-Gateway using base64 decoding, arriving at iss , attaching the .well-known/openid-configuration.
3. Use of middleware for the above point 2.
4. API-Gateway on nodejs
5. NestJS as background service/API.
6. Demo of CRUD using POSTMAN using authorization token.

Flow diagram for the project