Daal Backend Developer Challenge: Wallet Microservice

Overview
We are seeking a talented backend developer and this is a test challange to create a microservice that manages user wallet data. This service will expose two APIs to interact with other microservices, handling user balances and transactions. Your implementation should be in Node.js, using the Nest.js framework and "**TypeScript**".

API Requirements
1. **GET /balance**
    - This endpoint should return the current balance of a user in JSON format.
    - **Input**: **user_id** (integer)
    - **Output**: JSON object containing the balance, e.g., **{"balance": 4000}**
2. **POST /money**
    - This endpoint should allow adding or subtracting money from a user's wallet and return a transaction reference number.
    - **Input**: JSON payload with **user_id** (integer) and **amount** (integer; negative for deductions)
    - **Output**: JSON object with the transaction reference ID, e.g., **{"reference_id": 123123123}**

Technical Considerations
- **Database**: Use PostgreSQL or MongoDB for data persistence. Ensure the database schema is well-designed to handle the requirements.
- **Testing**: Include at least six test cases to demonstrate your understanding of testing methodologies and ensure the reliability of your service.
- **Transaction Logs**: All transactions should be logged with sufficient detail for auditing purposes.
- **Daily Totals**: Calculate and log the total amount of transactions processed each day.

Additional Instructions
- **Data Types**: Make sure to use appropriate field types in the database to match the input/output specifications.
- Although not required, containerizing the project with Docker and exploring communication protocols beyond REST (such as gRPC or GraphQL) will be viewed favorably.

Evaluation Criteria
- **Functionality**: The service must meet the specified requirements.
- **Code Quality**: Your code should be clean, well-organized, and easy to read.
- **Testing**: Adequate test coverage is essential for reliable software.
- **Documentation**: Include clear instructions on how to set up and run your project, as well as any design decisions and assumptions you made.

Submission Guidelines
Please provide a GitHub repository link containing your project. Ensure the repository is public or shareable with specific accounts (https://github.com/farshidbeheshti). Include a README with setup instructions and any other documentation you deem necessary.

Daal.co