ASP.NET Core 8 Backend Development Task

Technical Assignment

Overview

Create an ASP.NET Core 8 application with REST API. You do not need to create a UI.

The application must use one of the following databases via code-first approach: PostgreSQL (preferred), MS SQL, or MySQL.

Database Design

1. Nodes of Independent Trees:

- Each node must belong to a single tree.
- All child nodes must belong to the same tree as their parent.
- Each node must have a mandatory name field.
- You may add any additional fields necessary for ensuring tree independence.

2. Exception Journal:

- Track all exceptions during REST API request processing.
- Each journal record must store:
  - Unique Event ID
  - Timestamp
  - All query/body parameters
  - Stack trace of the exception

API Requirements

Your REST API should replicate (as closely as possible) the structure provided in the example Swagger file (see attached Swagger JSON). Improvements are welcome, as long as the core concept is maintained.

Exception Handling

Define a custom exception class named SecureException.

If a SecureException (or its subclass) is thrown during request processing:

- Store all details in the journal
- Respond with HTTP 500 and this format:

{"type": "name of exception", "id": "id of event", "data": {"message": "message of exception"}}

Example:

{"type": "Secure", "id": "638136064526554554", "data": {"message": "You have to delete all children nodes first"}}

For all other exceptions:

- Log full details in the journal
- Respond with HTTP 500 and:

{"type": "Exception", "id": "id of event", "data": {"message": "Internal server error ID = id of event"}}

Example:

{"type": "Exception", "id": "638136064187111634", "data": {"message": "Internal server error ID = 638136064187111634"}

Authentication(optional)

Your REST API can be authenticated via jwt token acquired by passing unique code to authentication API.