Test Case

Our goal is to develop a robust navigation module that empowers users to save and manage their journeys within our application. The module should collect essential information about each journey, including the starting location and time, arrival location and time, transportation type, and route distance.

User Functionality: For regular users, the following functionalities should be implemented:

1. **Register/Login**: Users can create an account or log in to access the navigation module.
2. **Create a Journey**: Users can create and save new journeys with all relevant details.
3. **Delete a Journey**: Users have the option to delete any of their saved journeys when necessary.
4. **Retrieve Journeys**: Users can retrieve specific journeys using their unique IDs or access a list of all their saved journeys.
5. **Daily Goal Achievement**: Users are eligible for a daily reward when their journey's total distance exceeds 20 km. The achievement status will be saved as a boolean value in the database.
6. **Log Out**: Users can securely log out of their accounts when they finish using the application.

Administrator Functionality: Administrators are granted additional capabilities to oversee and manage the system effectively:

1. **Journey Filtering**: Administrators can filter journeys based on various parameters, such as users, transportation types, and start/arrival dates.
2. **Monthly Route Distance Display**: The system will display the total route distance covered by users on a monthly basis to provide insights into user activity.

Technical Requirements:

To ensure a professional and scalable solution, the development must adhere to the following guidelines:

Backend: The backend should be designed and structured to accommodate both mobile and web consumers effectively. It must be scalable to handle future expansion.

Technology Stack: Utilize the latest .NET versions, such as .NET 6 or .NET 7, to leverage advanced features and security enhancements.

Dockerization: The web APIs and database should be Dockerized to facilitate easy launch across various environments.

Database: Implement a well-designed database structure, which can be SQL or NoSQL based on the specific application requirements.

Validation and Exception Handling: Implement comprehensive data validation to ensure data integrity and robust exception handling to provide a smooth user experience.

Authentication and Authorization: Implement secure authentication and authorization mechanisms to protect sensitive user data and ensure proper access control.

Unit Testing: Develop and implement unit tests to ensure reliable functionality and minimize bugs.

Code Quality: Maintain high code quality standards by following best practices and adhering to coding conventions.

Swagger Documentation: Provide comprehensive Swagger documentation for all API endpoints to aid developers in understanding and using the APIs effectively.

Bonus Features:

While not mandatory, the following bonus features will add significant value to the project:

1. **Caching Implementation**: Implement caching techniques to optimize data retrieval and improve system performance.

Deliverables:

The final deliverable should be the complete source code hosted in an online repository for easy access and collaboration.

Conclusion: By adhering to these professional guidelines, we aim to create a reliable, scalable, and secure navigation module that provides users with a seamless experience while managing their journeys.