**Asset Tracking System with Ownership Details**

**Overview:**

Develop a Spring Boot REST API for managing assets and tracking their ownership details. Implement a bidirectional one-to-one relationship between assets and their respective owners, enabling users to view asset information along with the details of the owner and vice versa.

**Functional Requirements:**

**Asset Entity:**

1.Define an Asset entity with the following attributes:

* id: int (asset ID)
* name: String (name of the asset)
* description: String (description of the asset)
* value: double (monetary value of the asset)

2. Implement a bidirectional one-to-one mapping with the Owner entity.

**Owner Entity:**

1.Create an Owner entity with the following attributes:

* id: int (owner ID)
* name: String (name of the owner)
* email: String (email address of the owner)
* address: String (address of the owner)

2.Implement a bidirectional one-to-one mapping with the Asset entity.

**Controller, Service, and Repository:**

1.**Controller**: Create a Controller class named "AssetController" to handle API requests related to assets and owners.

2.**Service:** Implement a Service class named "AssetService" to handle business logic for managing assets and owners.

3.**Repository**: Create Repository interfaces named "AssetRepository" and "OwnerRepository" to interact with the database using Spring Data JPA.

**Endpoints:**

Develop RESTful API endpoints for managing assets and owners.

**Asset Endpoints:**

* POST - "/assets" --> Create a new asset.
* GET - "/assets/{id}" --> Retrieve details of a specific asset by ID, including ownership details.
* PUT - "/assets/{id}" --> Update details of a specific asset by ID.
* DELETE - "/assets/{id}" --> Delete a specific asset by ID.

**Owner Endpoints:**

* POST - "/owners" --> Create a new owner.
* GET - "/owners/{id}" --> Retrieve details of a specific owner by ID, including asset details.
* PUT - "/owners/{id}" --> Update details of a specific owner by ID.
* DELETE - "/owners/{id}" --> Delete an owner by ID.

**Additional Notes:**

* Implement bidirectional one-to-one mapping between the Asset and Owner entities to establish the relationship.
* Ensure cascading behavior for save and delete operations to maintain data consistency.
* Configure database connection details in the application.properties file.
* API Endpoint: localhost:8080