Problem statement

You are requested to create REST API for asset management system. You should expose CRUD operations for Asset and User models (or tables). Also you should be able to buy/return asset for specific user.

```
class Asset{
        int asset_id; //automatic unique sequence number (primary key)
        String asset_name;
        String asset_type;
        int quantity;
}

class User{
        int user_id; //automatic unique sequence number (primary key)
        String username;
}

class AssetHistory{
        int asset_id;
        int user_id;
        int quantity;
        String type; // purchase/return
        LocalDate createdOn;
}
```

The list of api operations required are,

Api Path	Type	Description
/assetmanagement/asset/{id}	GET	Get asset by id
/assetmanagement/asset/all	GET	Get all assets from the table
/assetmanagement/asset/{id}	PUT	Update asset by id
/assetmanagement/asset/	POST	Create new asset
/assetmanagement/asset/{id}	DELETE	Delete asset by id
/assetmanagement/user/{id}	GET	Get user by id
/assetmanagement/user/all	GET	Get all users from the table
/assetmanagement/user/{id}	PUT	Update user by id
/assetmanagement/user/{id}	DELETE	Delete user by id
/assetmanagement/user/	POST	Create new user
/assetmanagement/asset/ purchase/{asset_id}/{user_id}/ {quantity}	PUT	This operation should, 1. select asset from asset table and user from user table 2. throw error message if asset or user record does not exists 3. throw error message if the required number of

		quantity is not available to allocate 4. Else, Add an entry in AssetHistory table with the details and reduce the quantity in Asset table
/assetmanagement/asset/return/{asset_id}/{user_id}/{quantity}	PUT	This operation should, 1. select asset from asset table and user from user table 2. throw error message if asset or user record does not exists 3. throw error message if the required number of quantity is not available to allocate 4. Else, Add an entry in AssetHistory table with the details and increase the quantity in Asset table

NOTE:

1. You are required to use,
* Springboot
* Maven
* SpringREST
* Spring JPA
For this above application.

2. You can choose the same database which you have gone through during your tutorial. I believe it is mysql.

3. You are free to choose your own coding style and approach as long as it works

4. Below are the things which are good to have but you can ignore this if don’t know what they are,
* Swagger for documentation
* Junit for unit test cases